REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
01-11-2017	Final Report	18-Jul-2014 - 17-Jul-2017

4. TITLE AND SUBTITLE
Final Report: Robust and High Order Computational Method for Parachute and Air Delivery and MAV System

5a. CONTRACT NUMBER
W911NF-14-1-0428

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
611102

6. AUTHORS

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
Research Foundation of SUNY at Stony Brc
W-5510 Melville Library

Stony Brook, NY 11794 -3362

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
63678-MA.14

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Xiaolin Li
UU	UU	UU	UU		19b. TELEPHONE NUMBER 631-632-8354

RPPR Final Report
as of 01-Nov-2017

Agency Code:

Proposal Number: 63678MA **Agreement Number: W911NF-14-1-0428**
INVESTIGATOR(S):

Name: Xiaolin Li
Email: xiaolin.li@stonybrook.edu
Phone Number: 6316328354
Principal: Y

Organization: **Research Foundation of SUNY at Stony Brook University**
Address: W-5510 Melville Library, Stony Brook, NY 117943362
Country: USA
DUNS Number: 804878247 EIN: 141368361
Report Date: 17-Oct-2017 Date Received: 01-Nov-2017
Final Report for Period Beginning 18-Jul-2014 and Ending 17-Jul-2017
Title: Robust and High Order Computational Method for Parachute and Air Delivery and MAV System
Begin Performance Period: 18-Jul-2014 **End Performance Period:** 17-Jul-2017
Report Term: 0-Other
Submitted By: Xiaolin Li Email: xiaolin.li@stonybrook.edu
Phone: (631) 632-8354
Distribution Statement: 1-Approved for public release; distribution is unlimited.

STEM Degrees: 5 **STEM Participants:** 4

Major Goals: The objective of this project it to study numerical algorithms and develop a computational platform forthe study of the dynamic system involving highly complex geometric interface immersed in the continuum medium. The success of the proposed research will have direct impact on many field operations of the Army and enhance the understanding of some very important equipments which the Army employs while performing its combat or humanitarian missions. These equipments include the parachute system for both personnel and cargo delivery, the micro air vehicle system for reconnaissance and remote sensing in hazardous environment, and the deceleration system for the re-entry vehicle in space missions. In the long range, we hope that the intellectual properties invented under this proposal will become an asset of the Army Research Laboratories for their numerical testing and simulation of these equipments in different environments.

Our goal is to establish a computational platform for the simulation of a parachute air deceleration system using the dual-stress spring mass model coupled with an incompressible fluid solver through the impulse method. Our approach to simulating the parachute system is based on the front tracking method. We designed the data structure representing the fabric surface of the canopy to allow for the application of the FronTier library in the simulation of the dynamic motion driven by the gravitational force of the payload and the fluid pressure differential at the fluid-fabric interface. The discretized fabric surface is a homogeneously triangulated surface mesh. Angular stiffness has been added to the simple spring mass model in order for the model to conform to both the Young's modulus and Poisson ratio of the fabric. This duel-stress spring-mass model is coupled with the incompressible fluid solver with turbulence and porosity models. We intend to build a realistic simulation tool by including physically validated collision handling between different components of the system. The software will be fully parallelized and can run efficiently on high performance computer system.

Accomplishments: We designed the data structure representing the fabric surface of the canopy to allow for the application of the FronTier library in the simulation of the dynamic motion driven by the gravitational force of the payload and the fluid pressure differential at the fluid-fabric interface. The discretized fabric surface is a homogeneously triangulated surface mesh. Angular stiffness has been added to the simple spring mass model in order for the model to conform to both the Young's modulus and Poisson ratio of the fabric.

The system of equations for the spring model is nonlinear and an analytical proof of its convergence is very difficult. Therefore, the proof of convergence is carried out through numerical mesh refinement. Using fixed initial and boundary conditions, the displacement, energy (kinetic energy and potential energy), and total length (2D) or area (3D) of the spring mesh are shown to be convergent under mesh refinement. Our numerical results show that the spring model is convergent under the conditions that the total mass of the fabric surface is kept constant and that

both the spring tensile stiffness and angular stiffness conform to Young's modulus and the Poisson ratio of the material.

Comparison of the drag force during inflation with the experimental data from Potvin shows agreement on the peak drag force at approximately 1.2 – 1.6 seconds after deployment. At this time, air from the lower side of the canopy rushes to fill the volume of the canopy causing expansion in both the vertical and horizontal directions. The ballooning canopy exerts the majority of the drag force on the stiffer string chords. Our simulation agrees with the experimental data in that there exists a time period during which the parachute canopy experiences a peak drag force that can be as large as 3-4 times that of the payload.

Two major components have been added to the code in order to achieve realistic simulations of the parachute inflation process: porosity and turbulence modeling. Due to the porous structure of fabric materials, a fraction of the freestream flow is able to pass through the canopy surface. This secondary flow stabilizes the parachute's descent and reduces drag force on the canopy. We carried out a distinct numerical study of the aerodynamic interactions of a porous fabric surface using the front tracking method combined with the ghost fluid method (GFM). In the front tracking framework, the fabric surface is treated as a dynamic elastic interface whose motion is driven by gravity, the surrounding fluid pressure and internal forces determined by the spring mass model. The surrounding air flow is described by the incompressible Navier-Stokes equations, which we solve numerically by the projection method and couple to the Ergun equation by imposing a pressure drop boundary condition at the interface. Since the projection method requires computing the pressure by solving the Poisson equation at each time step, it is convenient to use the GFM technique to allow the discontinuity in the pressure at the fluid-fabric interface to be handled with the Poisson solver.

Validation of the porosity model in the simulation was accomplished by comparing the drag forces with those observed in the impermeable cases. The drag forces and drag coefficients were calculated by varying the freestream velocity at the inlet. To study the effects of the porosity on the parachute system, we fix the inlet velocity while gradually increasing the permeability of the parachute canopy. Although the porosity is not explicitly defined in our model, its influence is captured by the parameters alpha (viscous porosity coefficient) and beta (inertial porosity coefficient) in the Ergun equation.

The hydrodynamic behavior of a turbulent incompressible fluid is governed by the Reynolds-Averaged Navier-Stokes (RANS) equations for the mean velocity and pressure. We consider the k-epsilon family of turbulence models, which automatically calculates the turbulence length scale. Our implementation differs from the standard k-epsilon model which is unable to capture the effects of smaller scales of motion due to its single turbulence length scale. In order to account for the different scales of motion, a mathematical technique known as the Re-Normalization Group (RNG) method is used to derive a turbulence model, similar to the standard one, resulting in a modified form of the epsilon equation.

Turbulence modeling makes a substantial difference in the simulation when a parachutist or cargo payload is included in the system. In either case, the payload is modeled as a rigid body in the system.
In the simulation of a C-9 personnel parachute with a parachutist, the numerical solution of the fluid flow near the parachute system shows that a turbulent wake appears as the fluid passes the parachutist. The descent speed of the parachute system is smaller in comparison to the speed observed in simulations in which the payload is a point mass. We believe this is due to the interaction between the parachutist and the fluid flow, which generates vorticity and increases the upward impulse on the parachute canopy. The form drag, caused by the pressure mismatch acting on the frontal area of the parachute presented to the flow, is greater in the laminar case than in the turbulent case. Although the frictional drag of the turbulent case is higher, the form drag is the dominant term in both cases.

Several collision handling features have also been added to the code. The collision resolving module for structures is also based on the front tracking framework. Using triangular meshes, the proximity and collision detection is broken down into two major cases: point vs. triangle and edge vs. edge. For example, when we detect a collision between a pair of triangles 15 tests are needed. Each point of one triangle is tested against the other triangle, and each edge of one triangle is tested against each edge of the other triangle. When suspension lines are included, the comparing units consist of not only point-triangle (triangle-triangle) and edge-edge, but also edge-triangle. There is one test for each edge-edge pair, and five tests for an edge-triangle pair. If rigid bodies are involved in a collision we apply the impact zone algorithm, which was initially proposed to deal with multiple collisions while maintaining collision consistency. The algorithm for the fabric surface and rigid body are similar, but there are some differences in the details of their implementations. The impulse acting on a mass point due to a collision consists of both an elastic impulse and an inelastic impulse. Our collision algorithm has been tested and verified for various

benchmark problems.

We developed our simulation platform on a parallel computing cluster consisting of one head node, 20 CPU (Central Processing Unit) nodes and one GPU (Graphic Processing Unit) node. Each node is populated by an Eight-Core Intel Xeon E5-2630v3 2.4GHz processor with Intel Hyper-Threading Technology. The GPU node has a 128 GB of RAM and a 240 GB SSD hard drive along with its own cloned operating system. The fluid solver uses multiple CPU cores; each is responsible for the solution in a single subdomain of the computational domain. To accelerate the calculations of the spring model we send the data to the GPU cores for massively vectorized processing in order to achieve the same level of speedup as the fluid solver. After the spring solver has been applied, the data is sent back to their corresponding subdomain data structures to provide the interface information needed by the fluid solver. By applying this technique of GPU accelerated computing to the entire parachute system, including both the parachute canopy and all of the suspension lines, we can achieve 16-21 times speedup compared to using just a single CPU node for the computation. We are in the process of porting the code into the DOD HPC platform.

Training Opportunities: Five graduate students have been awarded PhD degrees with thesis topics directly or partially related to this project. We have also participated in the ARO High School Apprenticeship Program (HSAP) and mentored six high school students in their summer internship.

Results Dissemination: Our research project has produced two publications in the Journal of Fluid and Structure, one publication in the AIAA journal, one in Communication in Computational Physics, along with several related publications in other journals. Two other papers have been submitted for publication and are currently under review. There have also been numerous conference presentations given by the Principal Investigator Xiaolin Li and the graduate students engaged in this research project. The FronTier code, developed with the support from this grant, has been sought and used for applications in several academic institutions and national laboratories including University of Connecticut Medical Center, University of Waterloo, Oak Ridge National Lab and Brookhaven National Lab.

Honors and Awards: Nothing to Report

Protocol Activity Status:

Technology Transfer: We are in the process of adapting our simulation system to the DOD HPC platform so that it can be used by Army and Airforce scientists for different applications.

PARTICIPANTS:

Participant Type: PD/PI
Participant: Xiaolin Li
Person Months Worked: 2.00 **Funding Support:**
Project Contribution:
International Collaboration:
International Travel:
National Academy Member: N
Other Collaborators:

Participant Type: Co-Investigator
Participant: Zheng Gao
Person Months Worked: 11.00 **Funding Support:**
Project Contribution:
International Collaboration:
International Travel:
National Academy Member: N
Other Collaborators:

Participant Type: Co-Investigator

RPPR Final Report
as of 01-Nov-2017

Participant: Xiaolei Chen
Person Months Worked: 10.00 **Funding Support:**
Project Contribution:
International Collaboration:
International Travel:
National Academy Member: N
Other Collaborators:

Participant Type: Co-Investigator
Participant: Tengbo Yang
Person Months Worked: 9.00 **Funding Support:**
Project Contribution:
International Collaboration:
International Travel:
National Academy Member: N
Other Collaborators:

Participant Type: Co-Investigator
Participant: Brandon Ballentine
Person Months Worked: 9.00 **Funding Support:**
Project Contribution:
International Collaboration:
International Travel:
National Academy Member: N
Other Collaborators:

DISSERTATIONS:

Publication Type: Thesis or Dissertation
Institution: Stony Brook University
Date Received: 15-Aug-2016 Completion Date: 8/1/16 10:07PM
Title: Modeling of Parachute Dynamics with GPU Enhanced Continuum Fabric Model and Front Tracking Method
Authors: Qiangqiang Shi
Acknowledged Federal Support: **Y**

Publication Type: Thesis or Dissertation
Institution: Stony Brook University
Date Received: 29-Oct-2017 Completion Date: 10/29/14 6:03PM
Title: Modeling of Parachute Dynamics with GPU Enhanced Continuum Fabric Model and Front Tracking Method
Authors: Qiangqiang Shi
Acknowledged Federal Support: **Y**

Publication Type: Thesis or Dissertation
Institution: Stony Brook University
Date Received: 29-Oct-2017 Completion Date: 5/29/17 6:06PM
Title: Numerical coupling and simulation of point-mass system with the turbulent fluid flow
Authors: Zheng Gao
Acknowledged Federal Support: **Y**

RPPR Final Report
as of 01-Nov-2017

Publication Type: Thesis or Dissertation
Institution: Stony Brook University
Date Received: 29-Oct-2017 Completion Date: 8/20/17 6:09PM
Title: Rigid Body Modeling in FronTier++ and Its Application to Parachute System Simulations
Authors: Xiaolei Chen
Acknowledged Federal Support: **Y**

Publication Type: Thesis or Dissertation
Institution: Stony Brook University
Date Received: 29-Oct-2017 Completion Date: 8/2/17 6:11PM
Title: A Numerical Study of Shear Instability in an Open Channel with Emergent Vegetation
Authors: Jingfan Qu
Acknowledged Federal Support: **Y**

Publication Type: Thesis or Dissertation
Institution: Stony Brook University
Date Received: 29-Oct-2017 Completion Date: 8/2/17 6:14PM
Title: Numerical Study of Reaction-Diffusion Systems using Front Tracking
Authors: Saurabh Gajanan Joglekar
Acknowledged Federal Support: **Y**

Program Director: Joseph Myers
Army Research Office

ARO: W911NF-14-1-0428

Final Report

Robust and High Order Computational Methods for Parachute Air Delivery and MAV Systems

Principal Investigators:
Xiaolin Li
University at Stony Brook

Reporting Period:
August 1, 2014 – July 31, 2017

Recipient:
Research Foundation
University at Stony Brook
Stony Brook, NY 11794-3366

1 Mathematical and Computational Objective

The mathematical objective of this project is the exploration of realistic models to numerically simulate the interaction between fluid and fabric surfaces, and to develop computational software for solving such problem. The technology developed from this project will be applied to the parachute deceleration system, airbag system and the Micro Air Vehicle (MAV) system. The success of this research will have significant impact on the design and optimization of some of the important equipments the Army employs in combat and humanitarian missions.

2 Parachute Modeling and Simulation

Parachute system is a complex system requiring detailed study of many aspects in computational fluid mechanics and computational structure mechanics. We have established a computational platform for the simulation of a parachute air deceleration system using the dual-stress spring mass model coupled with an incompressible fluid solver through the impulse method.

Here we report the accomplishment of this project during the period starting from August 1, 2014 to July 31, 2017. We outline the progress in the following areas:

1. modeling of fabric surface,
2. coupling with fluid solver,
3. modeling of turbulence,
4. modeling of parachute porosity,
5. collision handling,
6. interaction with cargo and parachutist,
7. parallelization of the software,
8. folding algorithm.

2.1 Modeling of Fabric Surface

Our approach to simulating the parachute system is based on the front tracking method. We designed the data structure representing the fabric surface of the canopy to allow for the application of the FronTier library in the simulation of the dynamic motion driven by the gravitational force of the payload and the fluid pressure differential at the fluid-fabric interface. The discretized fabric surface is a homogeneously triangulated surface mesh. Angular stiffness has been added to the simple spring mass model in order for the model to conform to both the Young's modulus and Poisson ratio of the fabric.

We applied a mesoscale model using the spring-mass system and strengthened it by the inclusion of both tensile stiffness and angular stiffness, a model proposed by Delingette in 2008 [8]. In this model, the energy $W(T_{\mathbf{X}_0})$ required to deform a single triangle $T_{\mathbf{X}_0}$ with vertices $\{\mathbf{X}_{10}, \mathbf{X}_{20}, \mathbf{X}_{30}\}$ into its new shape $T_{\mathbf{X}}$ with vertices $\{\mathbf{X}_{1,}, \mathbf{X}_2, \mathbf{X}_3\}$ consists of two parts:

- the energy of three tensile springs that prevent edges from stretching or compressing;
- the energy that prevent the change of each of the three angles.

For a triangle in equilibrium $T_{\mathbf{X}_0}$, the initial states are given by area $A_{\mathbf{X}_0}$, angles α_i, and lengths l_{ij}^0 $(i, j \in 1, 2, 3, j \neq i)$ in equilibrium while $A_{\mathbf{X}}$, β_i and l_{ij} denote the area, angles, and lengths of the deformed triangle $T_{\mathbf{X}}$ respectively.

The potential energy is given [8] as

$$W(T_{\mathbf{X}_0}) = \sum_{\substack{i=1 \\ j=(i+1) \ mod \ 3}}^{3} \frac{1}{2} \kappa_{ij}^{T_{\mathbf{X}_0}} (dl_{ij})^2 + \sum_{\substack{i=1 \\ j=(i+1) \ mod \ 3 \\ k=(i+2) \ mod \ 3}}^{3} \gamma_i^{T_{\mathbf{X}_0}} dl_{ij} dl_{ik}$$

where

$$\kappa_{ij}^{T_{\mathbf{X}_0}} = \frac{(l_{ij}^0)^2 (2 \cot^2 \alpha_k (\lambda + \mu) + \mu)}{8 A_{\mathbf{X}_0}}$$

is the tensile stiffness, and

$$\gamma_i^{T_{\mathbf{X}_0}} = \frac{l_{ik}^0 l_{ij}^0 (2 \cot \alpha_j \cot \alpha_k (\lambda + \mu) - \mu)}{8 A_{\mathbf{X}_0}}$$

is the angular stiffness, and γ and μ are the Lamé coefficients of the material. These coefficients are related to the two macroscopic physical parameters defined in planar elasticity for a membrane, that is, the Young's modulus E and the Poisson ratio ν by [12]:

$$\lambda = \frac{E\nu}{1 - \nu^2} \quad \text{and} \quad \mu = \frac{E(1 - \nu)}{1 - \nu^2}.$$

Young's modulus quantifies the stiffness of the material, whereas the Poisson ratio characterizes the material compressibility. The resulting force on each vertex of the spring system can be derived as

$$\begin{aligned} \mathbf{F}_{ij} &= ((\kappa_{ij}^{T_1} + \kappa_{ij}^{T_2}) dl_{ij} + (\gamma_i^{T_1} dl_{im} + \gamma_j^{T_1} dl_{jm} + \gamma_i^{T_2} dl_{in} + \gamma_j^{T_2} dl_{jn})) \mathbf{e}_{ij} \\ &= \tilde{\kappa}_{ij} dl_{ij} \mathbf{e}_{ij} + \tilde{\gamma}_{ij} dl_{ij} \mathbf{e}_{ij} \end{aligned} \tag{1}$$

where $\tilde{\kappa}_{ij} = \kappa_{ij}^{T_1} + \kappa_{ij}^{T_2}$, $\tilde{\gamma}_{ij} = (\gamma_i^{T_1} dl_{im} + \gamma_j^{T_1} dl_{jm} + \gamma_i^{T_2} dl_{in} + \gamma_j^{T_2} dl_{jn})/dl_{ij}$ and $\mathbf{e}_{ij}$ is the unit vector from $\mathbf{X}_i$ to $\mathbf{X}_j$.

The system of equations for the spring model is nonlinear and an analytical proof of its convergence is very difficult. Therefore, the proof of convergence is carried out here through

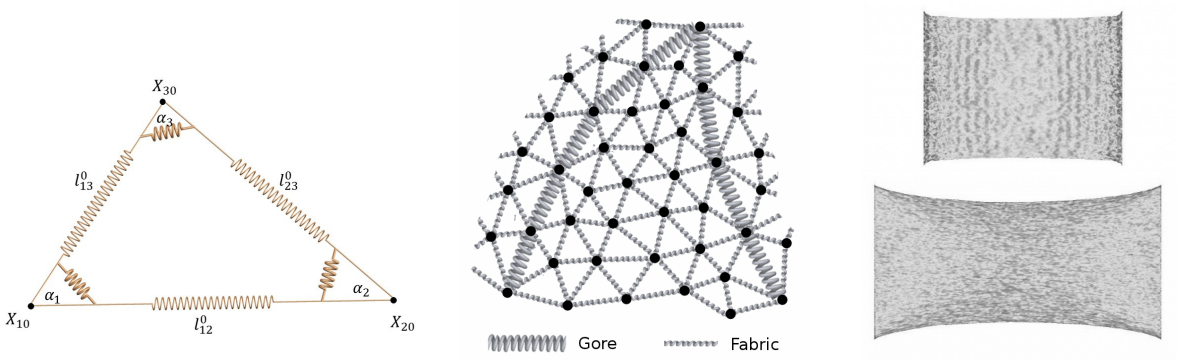


Figure 1: Dual-stress fabric model. Both tensile stiffness and angular stiffness are set for each triangle. Gore boundaries are modeled by springs with higher tensile stiffness. The model conforms with both Young's modulus and Poisson ratio of the fabric material. The right plot shows the stretching of the fabric surface.

numerical mesh refinement. Using fixed initial and boundary conditions, the displacement, energy (kinetic energy and potential energy), and total length (2D) or area (3D) of the spring mesh are shown to be convergent under mesh refinement. It is not surprising that the order of convergence is first order since the equations governing each vertex of the spring mesh involve only its immediate neighbors. Consequently, the convergence is not ideal when the surface becomes compressed or wrinkled. Our numerical results show that the spring model is convergent under the conditions that the total mass of the fabric surface is kept constant and that both the spring tensile stiffness and angular stiffness conform to Young's modulus and the Poisson ratio of the material.

Numerical implementation and testing have shown that such model is robust and conforms with both the Young modulus and Poisson ratio for a given fabric material. The left plot of Figure 1 shows the duel-stress fabric model.

2.2 Coupling with Fluid Solver

The fluid flow is computed by solving the incompressible Navier-Stokes equation using the finite difference method with high order coupling at the boundary and interior interface. To simplify the interaction between the fabric surface such as the parachute canopy and the fluid and to avoid harmful damping to the external driving force, we use a special method that separates the impulse from the external driving forces including the gravity and fluid pressure, and the impulse due to internal force each mass point receives from its neighboring points.

$$\mathbf{I}_i^c = \mathbf{I}_{gi}^c + \mathbf{I}_{pi}^c + \mathbf{I}_{si}^c \tag{2}$$

In our method, the external impulse (due to gravity and pressure) is time integrated for each mass point, that is

$$\mathbf{I}_{gi} = \int_0^t m\mathbf{g}dt \tag{3}$$

for both canopy and string cords mass points and

$$\mathbf{I}_{pi} = \int_0^t \sigma(p^- - p^+)\mathbf{n}dt \tag{4}$$

for canopy points only, where p^- and p^+ are the pressure on lower and upper sides of the parachute canopy, σ is the mass density of canopy per unit area, and $\mathbf{n}$ is the unit normal vector pointing from lower to upper side of the canopy.

We solve the incompressible Navier-Stokes equation using the the projection method [6, 4] and couple the fluid equation with the canopy surface through the impulse method [18].

Comparison of the drag force during inflation with the experimental data from Potvin shows agreement on the peak drag force at approximately 1.2 - 1.6 seconds after deployment. At this time, air from the lower side of the canopy rushes to fill the volume of the canopy causing expansion in both the vertical and horizontal directions. The ballooning canopy exerts the majority of the drag force on the stiffer string chords. Our simulation agrees with the experimental data in that there exists a time period during which the parachute canopy experiences a peak drag force that can be as large as 3-4 times that of the payload. However, an oscillatory variation in the drag force, or an aftershock, is exhibited that is not present in the data.

2.3 Turbulence Model

Due to large Reynolds number of the problem (which is in the magnitude of several millions for a full-scale parachute canopy[17]), turbulence model is needed. We apply the eddy viscosity models based on the Reynolds Averaged Navier-Stokes (RANS) equation. The hydrodynamic behavior of a turbulent incompressible fluid is governed by the RANS equations for the mean velocity $\mathbf{u}$ and pressure p

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} &= -\nabla p + \nabla\cdot((\nu+\nu_T)[\nabla\mathbf{u} + \nabla\mathbf{u}^T]) \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned} \tag{5}$$

where ν is the kinematic viscosity and ν_T is the turbulent eddy viscosity which approximates the turbulent fluctuations.

We compared two eddy viscosity models, the Baldwin-Lomax model and the $k-\varepsilon$ model. In the Baldwin-Lomax [2] algebraic model, $\nu_T = l_m^2\Omega$, where l_m is the mixing length specified with a damping function and Ω is the modulus of the mean rate of rotation tensor. The Baldwin-Lomax model is reliable since it seldom produces completely unphysical results. Therefore, we adopt this model as the first step to test the effect of turbulence on the

fluid flow surrounding the parachute canopy. However, this simple model tends to over-predict separation regions, thus not suitable for the region around parachute canopies. A more realistic one is the RANS-based turbulence model, or the $k-\varepsilon$ model family, which automatically calculate the turbulence length scale [29]. In standard $k-\varepsilon$ model, the eddy viscosity is defined as

$$\nu_T = C_\mu \frac{k^2}{\varepsilon}, \tag{6}$$

where k is the turbulence kinetic energy and ε is the dissipation rate. To compute k and ε, two additional convection-diffusion-reaction equations are needed:

$$\frac{\partial k}{\partial t} + \nabla \cdot (k\mathbf{u} - (\nu + \frac{\nu_T}{\delta_k})\nabla k) = P_k - \varepsilon \tag{7}$$

$$\frac{\partial \varepsilon}{\partial t} + \nabla \cdot (\varepsilon\mathbf{u} - (\nu + \frac{\nu_T}{\delta_\varepsilon})\nabla \varepsilon) = \frac{\varepsilon}{k}(C_1 P_k - C_2 \varepsilon) \tag{8}$$

where $P_k = \frac{\nu_T}{2}|\nabla\mathbf{u} + \nabla\mathbf{u}^T|^2$ is the production of turbulent kinetic energy. For the standard $k-\varepsilon$ model, the default values of the involved empirical constants are: $C_\mu = 0.09$, $C_1 = 1.44$, $C_2 = 1.92$, $\delta_k = 1.0$, $\delta_\varepsilon = 1.3$. Although simple and efficient, the standard model is unable to capture the effects of smaller scales of motion due to its single turbulence length scale. In order to account for the different scales of motion, a mathematical technique called Re-Normalization Group (RNG) method [31] is used to derive a turbulence model similar to the standard one, resulting in a modified form of the ε equation:

$$\frac{\partial \varepsilon}{\partial t} + \nabla \cdot (\varepsilon\mathbf{u} - (\nu + \frac{\nu_T}{\delta_\varepsilon})\nabla \varepsilon) = \frac{\varepsilon}{k}(C_1 P_k - C_2^* \varepsilon) \tag{9}$$

$$C_2^* = C_2 + \frac{C_\mu \eta^3 (1 - \eta/\eta_0)}{1 + \beta\eta^3} \tag{10}$$

$\eta = kS/\varepsilon$, S is the modulus of the mean rate of strain tensor. The coefficients are derived explicitly in the RNG procedure and for completeness, we list here as: $C_\mu = 0.0845$, $C_1 = 1.42$, $C_2 = 1.68$, $\delta_k = 0.7194$, $\delta_\varepsilon = 0.7194$.

The implementation of $k-\varepsilon$ model makes no difference in solving other convection-diffusion-reaction equations, except two distinctions. First, one must specify an appropriate boundary condition at reflecting wall for the velocity and two turbulence parameters. Due to extremely high Reynolds number in parachute simulation, it is worthwhile using the wall functions to bridge the viscosity-affected region and the fully-turbulent region in order to avoid the need for high resolution of strong velocity gradients [20]. Another task is to avoid loss of positivity of k and ε due to computational errors. This can be achieved by keeping the coefficients positive for the linearized equations without changing any primitive variables [21]. Figure 2 displays the viscosity and velocity streamline computed with RNG $k-\varepsilon$ model.

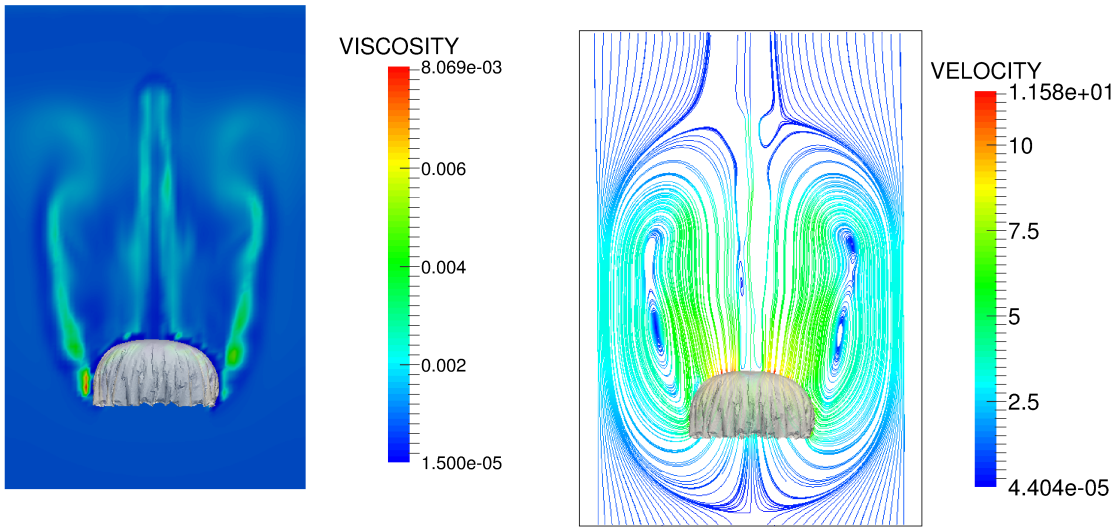


Figure 2: Viscosity and streamline around parachute computed by RNG $k-\varepsilon$ model

2.4 Porosity Modelling

The permeability of fabric membrane is modelled using Darcy's law through a thin layer. We apply the Ghost Fluid Method (GFM) to compute the pressure jump across the interface. We implemented the porosity to the parachute and measured its effects on the drag force during parachute inflation.

The fluid velocity $\mathbf{u} = (u, v, w)$ and pressure p are computed by the Navier-Stokes equation:

$$\begin{aligned} \rho(\mathbf{u}_t + \mathbf{u} \cdot \nabla \mathbf{u}) &= -\nabla p + \mu \Delta \mathbf{u} \\ \nabla \cdot u &= 0 \end{aligned} \tag{11}$$

where μ and ρ are dynamic viscosity and density of the fluid respectively. The fabric porosity is implemented using the pressure jump condition at the interface:

$$\begin{aligned} [p]_\Gamma &= r_\Gamma \mathbf{u} \cdot \mathbf{n} \\ [\partial_n p]_\Gamma &= 0 \\ [u]_\Gamma &= 0 \end{aligned} \tag{12}$$

where $[p]_\Gamma = p^+ - p^-$ is the pressure jump across the interface Γ, p^+ and p^- are the pressure on side Ω^+ and Ω^- sides respectively, r_Γ is the interface resistance which can be determined either from Darcy's law for laminar flow or Ergun's law for turbulence flow, and $\mathbf{n}$ is the local unit normal to the interface. The sign of the subdomains is decided by the normal vector which points from Ω^+ to Ω^-. In our application of parachute simulation, the interface is an open surface, which means Ω^+ and Ω^- are connected. However, this will not affect

the model we describe here. We applied the painting method to separate Ω^+ and Ω^- in the close neighborhood of the canopy.

The interaction between fluid and structure is handled by *FronTier++* library. The jump condition (13) is derived by coupling the GFM with finite difference projection scheme. Here we will mainly focus on the pressure-Poisson version of projection method since the jump condition can be directly applied to the Poisson equation.

The jump condition is considered when discretizing the pressure in the projection method for Navier-Stokes equation. Since the derivative is not defined across the interfaces, therefore, the techniques proposed in [22] for the variable coefficient Poisson equation are used for the discrete Laplace operator and the gradient operator. In the front tracking framework, we directly apply the local normal vector to determine the sign of a cell in the computational domain since the interface is explicitly represented by connected marker points in triangulated mesh. For example in Figure 3, when discretizing at point A along x direction, the sign of the cell center is defined by the sign of $\Phi_A = \mathbf{v}_A \cdot \mathbf{n}$. Therefore, the sign of a cell center is allowed to vary when discretizing along different directions. Applying technique in [22], we obtain

$$\Delta_h p^{n+1/2} = \nabla_h \cdot \mathbf{u}^*/\Delta t + F^x + F^y + F^z \tag{13}$$

where $F^x = F^W + F^E$, $F^y = F^N + F^S$ and $F^z = F^T + F^B$.

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t(\nabla_h p^{n+1/2} + G^x + G^y + G^z) \tag{14}$$

where $G^x = G^W + G^E$, $G^y = G^N + G^S$ and $G^z = G^T + G^B$. For illustration, we present the formulation of F^W, F^E, G^W, G^E, which only includes the information at X_i, $X_{i\pm1}$. The rest of the cases can be obtained similarly by the techniques introduced in [22]. If $\Phi_i \cdot \Phi_{i\pm1} \geq 0$, that is no interface exists between point X_i and $X_{i\pm1}$, then we have $F^W = G^W = 0$ and $F^E = G^E = 0$. In the situation of $\Phi_i \cdot \Phi_{i\pm1} < 0$

$$F^W = \begin{cases} r_\Gamma \mathbf{u}^n \cdot \mathbf{n}/\Delta x^2, & \Phi_i < 0 \text{ and } \Phi_{i-1} > 0 \\ -r_\Gamma \mathbf{u}^n \cdot \mathbf{n}/\Delta x^2, & \Phi_i > 0 \text{ and } \Phi_{i-1} < 0 \end{cases} \tag{15}$$

$$G^W = \begin{cases} r_\Gamma \mathbf{u}^n \cdot \mathbf{n}/\Delta x, & \Phi_i < 0 \text{ and } \Phi_{i-1} > 0 \\ -r_\Gamma \mathbf{u}^n \cdot \mathbf{n}/\Delta x, & \Phi_i > 0 \text{ and } \Phi_{i-1} < 0 \end{cases} \tag{16}$$

$$F^E = \begin{cases} r_\Gamma \mathbf{u}^n \cdot \mathbf{n}/\Delta x^2, & \Phi_i < 0 \text{ and } \Phi_{i+1} > 0 \\ -r_\Gamma \mathbf{u}^n \cdot \mathbf{n}/\Delta x^2, & \Phi_i > 0 \text{ and } \Phi_{i+1} < 0 \end{cases} \tag{17}$$

$$G^E = \begin{cases} -r_\Gamma \mathbf{u}^n \cdot \mathbf{n}/\Delta x, & \Phi_i < 0 \text{ and } \Phi_{i+1} > 0 \\ r_\Gamma \mathbf{u}^n \cdot \mathbf{n}/\Delta x, & \Phi_i > 0 \text{ and } \Phi_{i+1} < 0 \end{cases} \tag{18}$$

We validated our model by calculating the permeability velocity and pressure drop across the fabric interface. A correct model should reproduce the parabolic relation between these

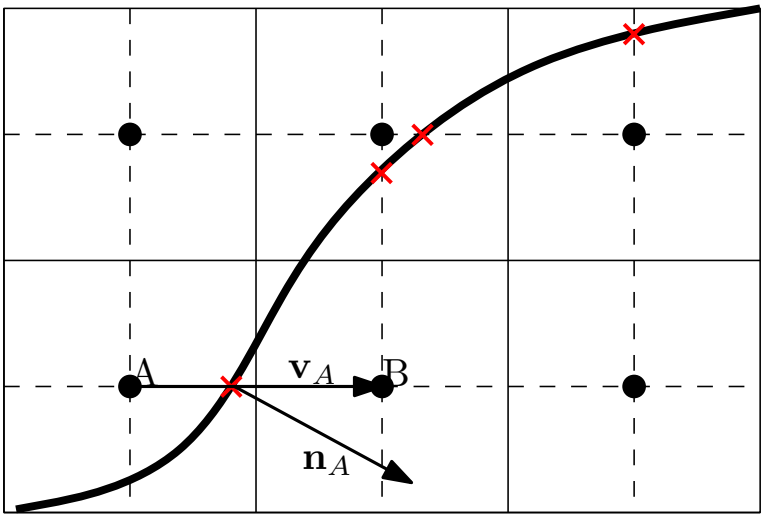


Figure 3: Illustration of computational grid and interface

two variables. We can then apply this model to the simulation of parachute and analyze the effects of porosity on the drag/lift ratio to the parachute.

Our test velocity magnitude and pressure with $U_{max} = 3m/s$ are shown in Figure 4 and the profiles of velocity and pressure at different locations are shown in Figure 5.

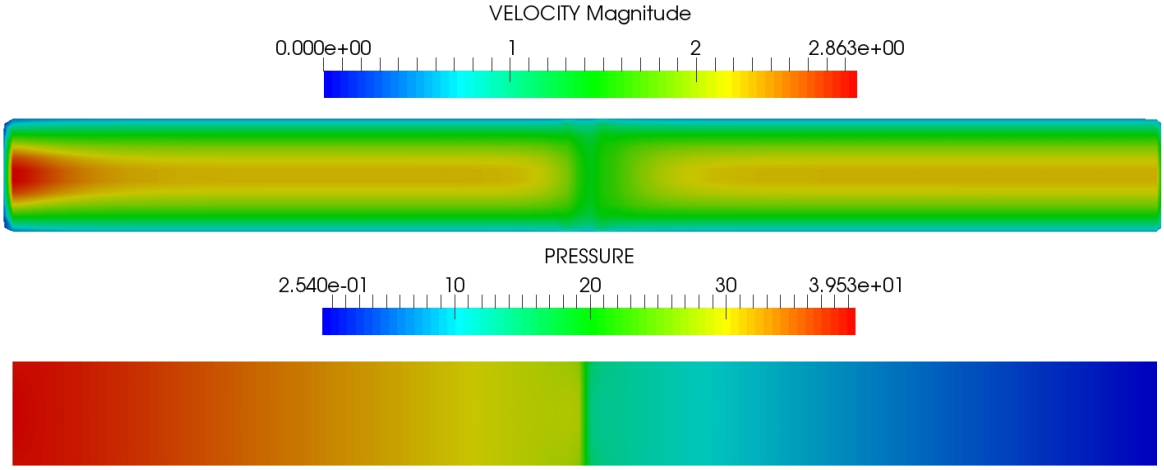


Figure 4: x-y cross-sectional plane of velocity (up) and pressure (bottom).

A full scale model is used to simulate the behavior of a real fabric in the channel flow. The boundary conditions of the computational domain are the same as the ones in the first test case except the domain size is now $30m \times 10m \times 10m$. The elastic fabric surface located at $x = 2m$ is modeled by the spring mesh introduced in [24], thus can be stretched or compressed due to the varying pressure drop across the interface. The fabric tested here resembles MIL-c-7020 type III fabric with density $533.77kgm^{-3}$, Young's modulus $0.4309Gpa$ and thickness

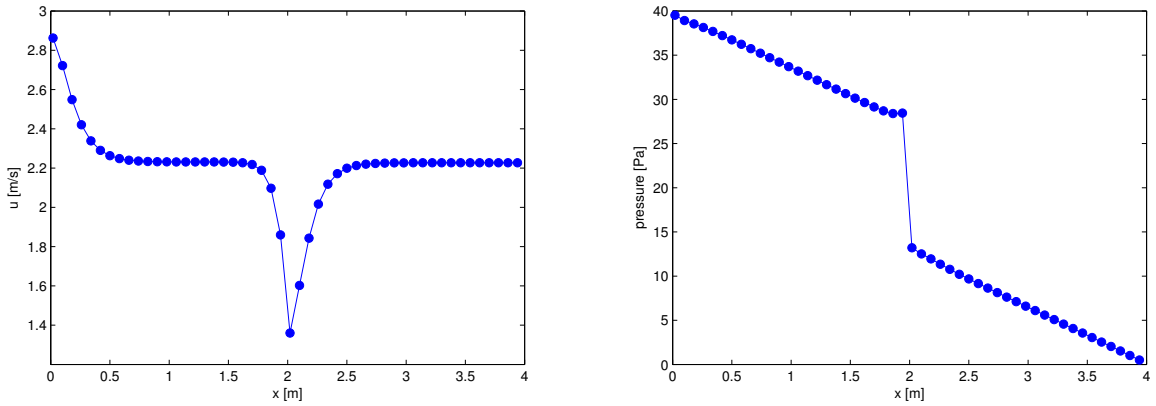


Figure 5: Velocity (left) and pressure (right) along central line.

$0.1016mm$. The values of viscous and inertial parameters are $\alpha = 162kgm^{-2}s^{-1}$ and $\beta = 48.82kgm^{-3}$. The velocity magnitude and streamline are displayed in Figure 6. To test the effects of porosity, the permeability velocity and pressure drop are recorded when the flow reaches steady state. The permeability velocity and pressure drop are measured by taking the average of velocity and pressure drop over the entire surface. By imposing different inflow velocity, a relation between these two variables can be recorded. The parachute system is affected by the porosity through in two ways. On one hand, the porosity model will reduce the drag force on the parachute surface by lowering the pressure difference. This can be verified by looking at the projected area which directly reflects the effects of drag force with varying porosity. On the other hand, the permeability of parachute canopy could significantly affect the aerodynamic field variables of the surrounding fluid such as pressure, flow velocity and vorticity. These, in turn, will impact the stability of the parachute system. We show this by plotting the enstrophy of the computational domain, see Figure 7.

2.5 Collision Handling

In fabric simulations, collision is inevitable. *FronTier++* has the capability to resolve the contact of surfaces between two fluids by merging or bifurcation. However, for structure interaction, the colliding components can neither merge nor bifurcate. Thus, a robust collision detection and different solution module is very important to make the numerical simulation physically accurate. Our collision algorithm is based on the combination of the fail-safe geometric collision method and the repulsion force method [**?**]. In addition, we extend this algorithm to not only the canopy, but also the suspension lines and the rigid bodies, either fixed or moving.

In Figure **??**, we show the procedures of the collision algorithm between a fabric structure and a static rigid body. For self-collision of the fabric surface and collision between fabric structure and a moving object, the algorithm is similar. The difference lies in how the impulse is applied and how the structures rebound back.

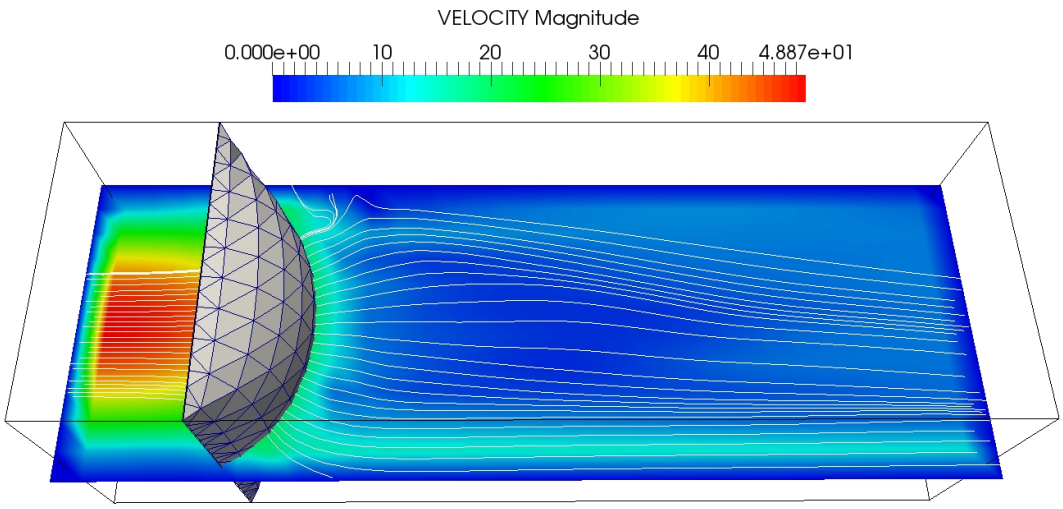


Figure 6: Velocity magnitude and streamline for $U_{max} = 10m/s$, mesh size is $30 \times 10 \times 10$

The following is a summary of the collision algorithm. The detection is carried out using the axis-aligned bounding box (AABB) tree [**?**, **?**]. The details on how the impulse is calculated and how the impact zone is determined are discussed in Sec. **??**.

- Pre-processing:
  - Starting at t^n with a time step Δt, record current positions $\mathbf{x}^n$ and velocities $\mathbf{v}^n$.
  - Propagate all structures to the candidate positions $\mathbf{x}^{can}$ with candidate velocities $\mathbf{v}^{can}$ at time t^{n+1}, and compute the average velocity during this time step, $\mathbf{v}^{ave} = (\mathbf{x}^{can} - \mathbf{x}^n)/\Delta t$.
  - Connect points on each rigid body as one impact zone.
- Detection and Handling:
  - Perform the proximity check for $\mathbf{x}^n$ and apply impulse and friction to update $\mathbf{v}^{ave}$.
  - Perform the collision check for propagation from $\mathbf{x}^n$ with velocity $\mathbf{v}^{ave}$, and resolve collisions by applying the impulse and rigid impact zones technique to update $\mathbf{v}^{ave}$.
  - Update the candidate position of all the points by $\mathbf{x}^{can} = \mathbf{x}^n + \Delta t \mathbf{v}^{ave}$.
  - If there is no more collision with respect to $\mathbf{x}^{can}$, continue to the post-processing; otherwise, repeat the detection and handling process.
- Post-processing:
  - Advance the $\mathbf{v}^{ave}$ to $\mathbf{v}^{can}$ for points involved in collision, and, meanwhile, update the center of mass and center of mass velocity of the moving rigid bodies.

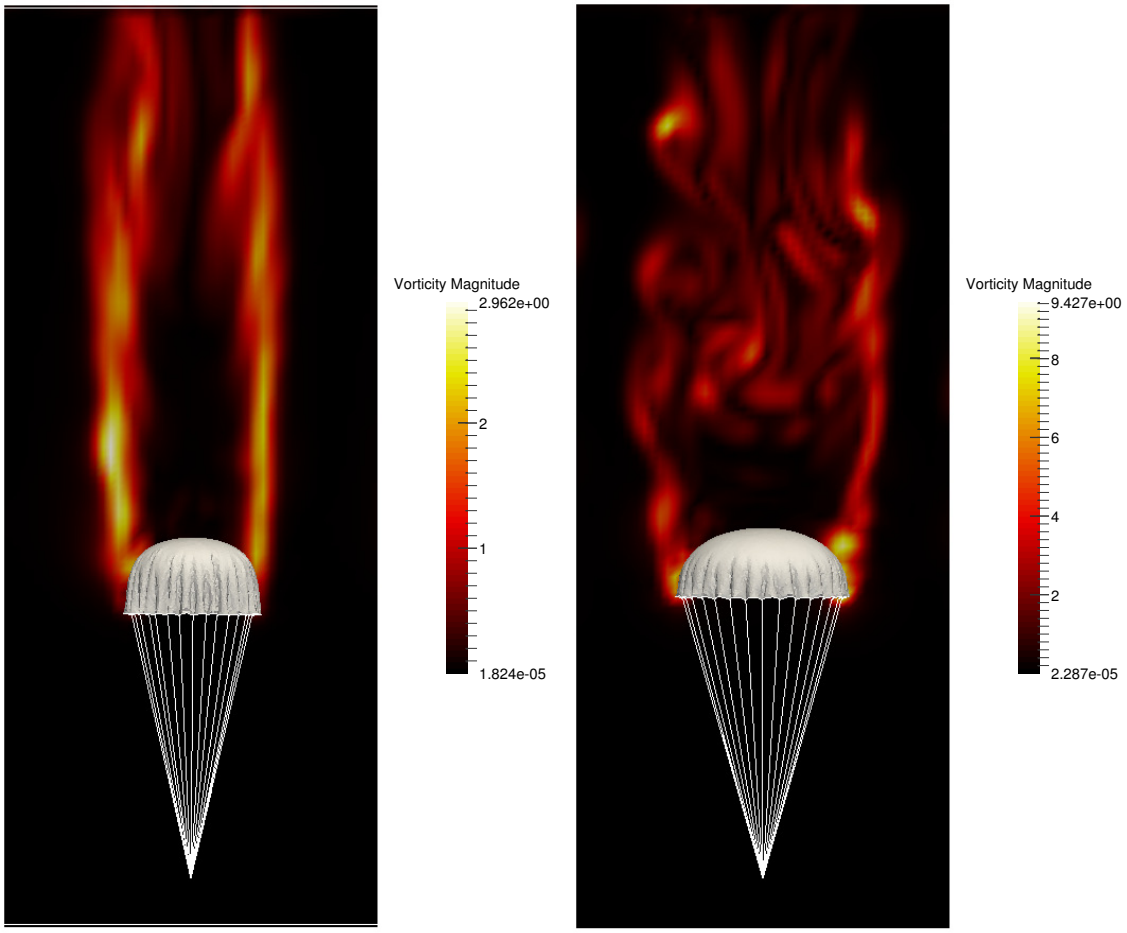


Figure 7: Qualitative comparison of vorticity magnitude $(1/s)$ of the same parachute with high porosity (left) and low porosity (right). The porosity coefficients for the left plot are $\alpha = 6.7 kgm^{-1}s^{-1}$, $\beta = 3.1 kgm^{-2}$ and for the right plot are $\alpha = 18.7 kgm^{-1}s^{-1}$, $\beta = 7.2 kgm^{-2}$.

– Continue to the next time step with $\mathbf{x}^{n+1} = \mathbf{x}^{can}$ and $\mathbf{v}^{n+1} = \mathbf{v}^{can}$ for all points.

Since one-dimensional collisions between two rigid bodies can be solved analytically and have a closed formula for velocities, we use this test case as a benchmark to check the accuracy of the algorithm. From Newton's second law, we have

$$\begin{aligned} \widetilde{v}_a &= \frac{C_R m_b (v_b - v_a) + m_a v_a + m_b v_b}{m_a + m_b} \\ \widetilde{v}_b &= \frac{C_R m_a (v_a - v_b) + m_a v_a + m_b v_b}{m_a + m_b} \end{aligned} \tag{19}$$

where C_R is the coefficient of restitution. $C_R = 1$ is for a fully elastic collision, and $C_R = 0$ for a fully inelastic collision.

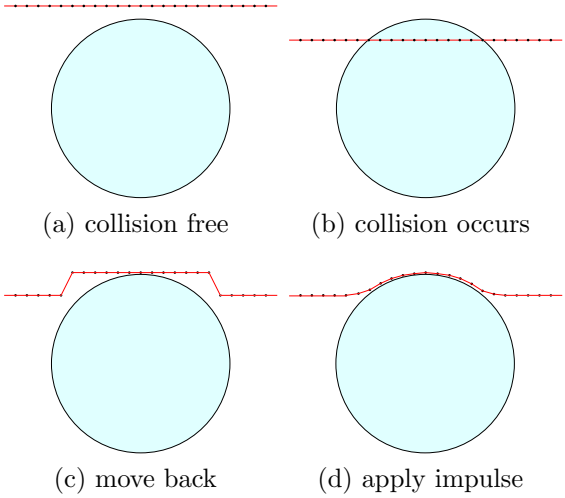


Figure 8: A demonstration of process how to resolve the collision. The red line represents the fabric structure; the light-blue disk represents an object; the points on the red line are the mass points in the spring model.

Consider two cuboids initially moving toward each with velocity $0.5m/s$ and -$0.5m/s$ in the x direction, with masses $1kg$ and $2kg$, respectively. Using Eq. 19, the speed after collision is

$$\widetilde{v}_a = -\frac{1}{6}m/s,\ \widetilde{v}_b = -\frac{1}{6}m/s,\ \text{with } C_R = 0;$$
$$\widetilde{v}_a = -\frac{5}{6}m/s,\ \widetilde{v}_b = \frac{1}{6}m/s,\ \ \text{with } C_R = 1.$$

Figure **??** demonstrates the numerical profiles of velocity in two extreme cases: $C_R = 0$ and $C_R = 1$. Clearly, it is consistent with the analytical solutions. Figure **??** Shows three states of the process for a fully inelastic case.

2.6 Modelling of Parachutists

The physical dimensions of the parachutist or cargo are usually smaller than the parachute canopy surface. However, when the ambient fluid flows around the parachutist, the dynamical behavior of the flow will result in different patterns characteristic to the flow Reynolds number. This produces turbulent flow which will reach the parachute canopy surface later and affect the motion of the canopy surface with random impulses. Therefore the inclusion of the rigid parachutist will make the numerical simulations of a parachute system more realistic. Figure **??** shows three examples of connecting the suspension lines to the parachutist or cargo.

In our model, the payload of the parachute can be either a mass point with no volume or a parachutist as rigid body with finite volume. In the former case, no interaction between the payload and the fluid is considered. Therefore, no turbulent flow will be generated in the

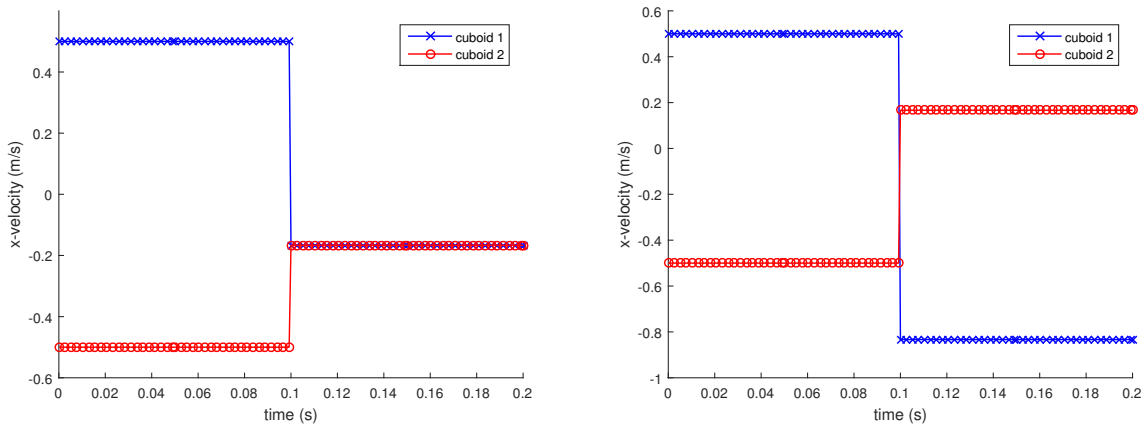
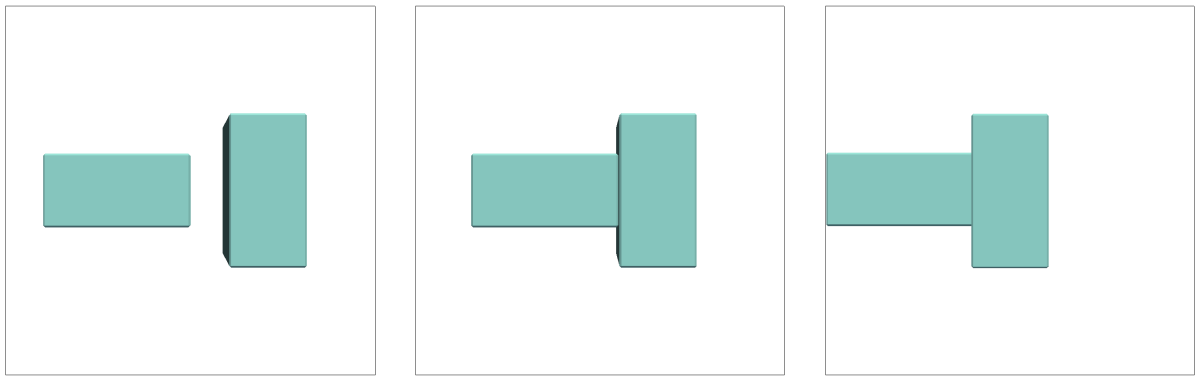
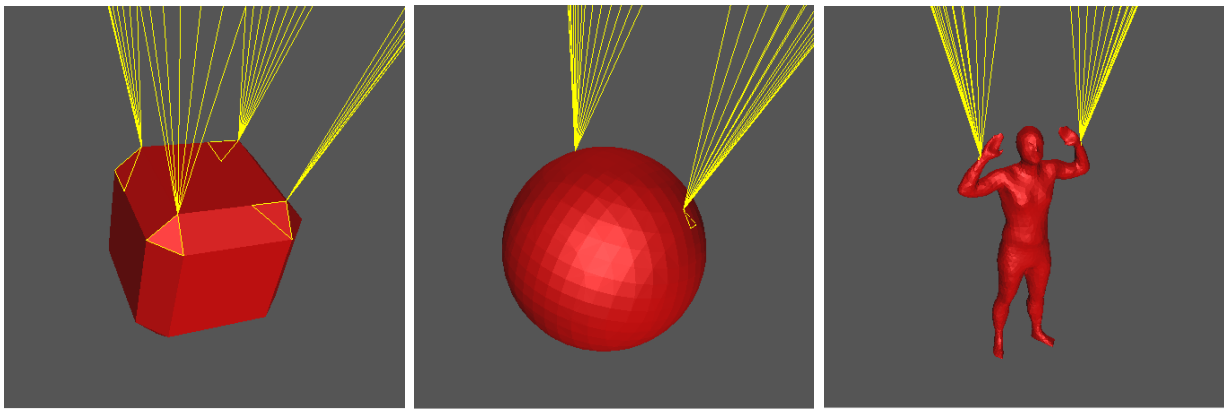


Figure 9: The velocity profile of the two cuboids. The left figure is the complete inelastic case ($C_R = 0$), and the right figure is the complete elastic case ($C_R = 1$).



(a) initial time = 0.0s (b) collision time = 0.1s (c) final time = 1.0s

Figure 10: Complete inelastic collision. The left figure is the initial state ($t = 0s$) and two cuboids are moving toward each other; the middle figure is when they collide ($t = 0.1s$), after which both move to the left with the same velocity; the right figure is the final state ($t = 1.0s$).



(a) cuboid cargo (b) spherical cargo (c) parachutist

Figure 11: Three examples of the connection amplification between the suspension lines and different types of payloads. Besides these three, *FronTier++* is capable of generating many other kinds of rigid bodies.

space between the canopy surface and the payload. However, in the latter case, the fluid-structure interaction must be considered, and there are two important aspects that need to be carefully taken into account.

- Calculation of force and torque on the parachutist.
  The force consists of three parts: the gravity, the force exerted by the fluid flow and the force from the suspension lines. The force and corresponding torque are then applied to the parachutist for translation and rotation movements.

  – The force exerted by the fluid flow can be broken into two parts: the force due to the pressure difference and the force due to friction (viscosity). In the parachute simulation, we only consider the first part as the dominant force.

$$F_p = \int_s p\hat{n} dA, \tag{20}$$

    where $\hat{n}$ is the normal direction to the surface with area dA, and p is the pressure at surface dA.

- Propagation of the rigid body boundary.
  Th propagation must maintain the geometric shape of the parachutist. This has a perfect match with the Lagrangian propagation of the front tracking method through *FronTier++* .

  – For the translational motion, we apply the acceleration due to the total force to the center of mass, and propagate each point of the parachutist in the same direction and same speed.

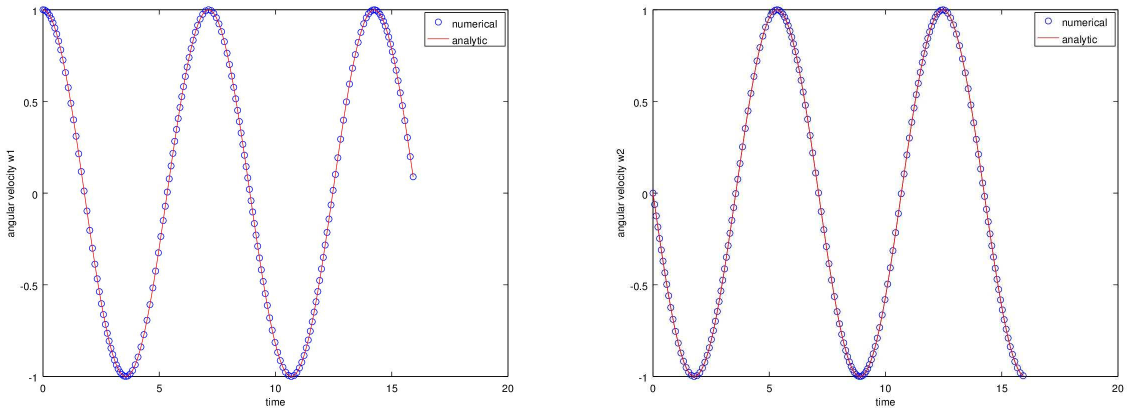


Figure 12: For the simple example with $I_1 = I_2 \neq I_3$, the left is the angular velocity in x direction (w_1), and the right is the angular velocity in y direction (w_2). The red line is the analytical value, while the blue circles are the numerical result.

– The rotational motion with respect to the center of mass is governed by Euler's equations, Eq. 21, of rigid body dynamics [**?**].

$$\begin{cases} I_1\dot{w}_1 - w_2w_3(I_2 - I_3) = \tau_1 \\ I_2\dot{w}_2 - w_3w_1(I_3 - I_1) = \tau_2 \\ I_3\dot{w}_3 - w_1w_2(I_1 - I_2) = \tau_3 \end{cases} \tag{21}$$

where, (I_1, I_2, I_3) is the principle momentum of inertia, (w_1, w_2, w_3) is the angular velocity, and (τ_1, τ_2, τ_3) is the outside torque exerted on the rigid body.

One benchmark test for the mechanics of rigid body motion is the rotation of a cone ($I_1 = I_2 \neq I_3$) with respect to its apex without gravity. Given initial angular velocity $\mathbf{w} = (0, 0, 1)$ and no fluid effect considered (vacuum), it has the analytic solution of the angular velocity.

$$\begin{cases} w_1 = \cos(\dfrac{I_3 - I_1}{I_1}t) \\ w_2 = \sin(\dfrac{I_3 - I_1}{I_1}t) \\ w_3 = 1 \end{cases} \tag{22}$$

And the numerical solution is consistent with the analytic solution, as shown in Figure **??**.

2.7 Folding Algorithm

Portable equipments with inflatable elastic or fabric surfaces, such as parachute and airbag systems, are usually packed into a folded state before deployment and inflation. The material

surfaces in these equipments may have different geometric shapes and folding patterns to serve for different functions of the equipment. Each folding pattern may contain several folding operations, such as a flat fold, long fold or an accordion fold, see Figure 13. Numerical preparation of the packed state is essential for a realistic simulation of the parachute inflation process. Four methods have been proposed by different investigators to solve the folding problem. The most notable one is the initial metric method (IMM), see [27, 33, 32]. Although the method is efficient, it is complex, fallible, and can fold only simple patterns. The second method is the simulation-based approach [13, 14, 28], which requires substantial preparation and computation time but can be applied to almost any folded pattern. Another method is a direct folding method proposed by Han Cheng, et al. [5]. In this method, Fluid Structure Interaction (FSI) based on ALE (Arbitrary Lagrangian Eulerian) method is used to help achieving the target folded state. The fourth approach is the origami-based folding algorithm [3, 26, 30, 10]. This method uses mathematical logic in designing the crease pattern and checking the foldability. When numerically implemented, it can create complicated folding structures. Anything the origami can make, such as the swan or other animation characters, can also be realized by using this method [15, 9]. Although this method is original designed for paper, we can add a post-procedure through the relaxation of spring-mass model for the fabric material.

Our folding sheet is on a trianglated mesh, therefore it requires the collective transformation of all the node point on the surface. Moreover, parachute canopy can be bent and stretched due to the physical properties of the fabric material while paper material can't. We then let the origami-folded pieces to relax through the spring-mass model by adding a small bending force to each pair of the triangles. The relaxation is detected and resolved for collisions after each time step. Our method can fold the fabric surface to many different states with high complexity efficiently.

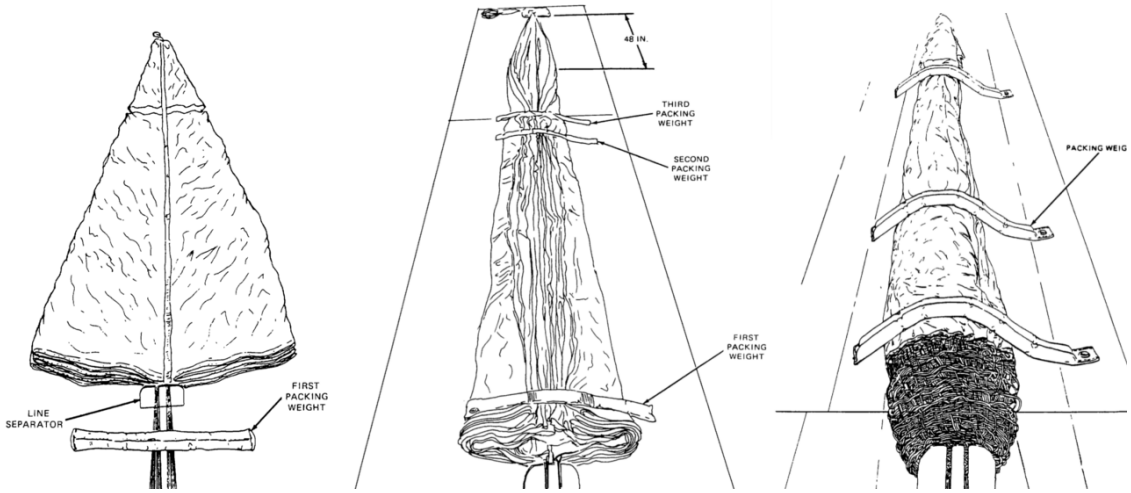


Figure 13: Snapshots of folding processes in [1], from left to right are flat fold, half-long fold and long fold.

2.7.1 Rigid origami algorithm

We consider one non-virtual vertex case. All other vertices are virtual, which means that they are on the boundary of the surface. For origami sequences, the first step is to determine the crease pattern. Crease pattern usually includes crease lines and the corresponding folding angles. Figure 14 shows the crease pattern of a corner box folding.

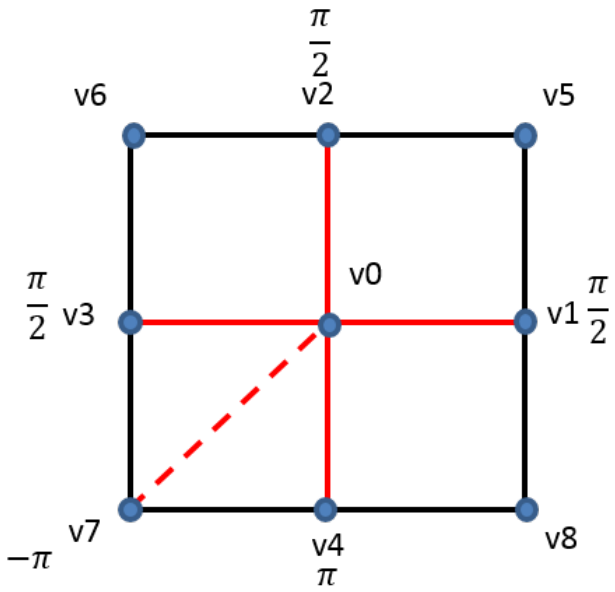


Figure 14: An example of a crease pattern. Solid lines corresponds to the positive folding angle and dashed lines correspond to the negative folding angle. v_0 is the only non-virtual vertex.

Not all crease patterns are foldable origami planes. An ill-designed crease pattern may cause stretches, rips and self-intersections at some steps the folding sequences, thus cannot be obtained from a plane surface. The necessary condition for a foldable crease pattern is given in [3]. Given the crease lines $l_1, l_2, ..., l_n$ ordered in the anti-clockwise direction, and the rotation matrices corresponding to the folding angles of the crease lines as $M_1, M_2, ..., M_n$, a foldable crease pattern must satisfy

$$M_1 M_2 ... M_n = I \tag{23}$$

where I denote the identity matrix. Although only a necessary one, this condition is very useful in determining the validity of a given crease pattern. This theorem plays an import role in our implementation of the algorithm.

Numerical implementation In our code, the randomized rigid origami folding algorithm with closure constraints proposed by Xi et al. [30] is used to design the origami procedures numerically. This algorithm is summarized as follows:

Given the initial configuration C_0, the target configuration C_T, current configuration C_c from C_0, and the initial weight W, search for step size S,

1. Get a random configuration C_r with each folding angle within $[-\pi, \pi]$.
2. Calculate searching direction $\boldsymbol{D} = (1 - W) \cdot C_r + W \cdot C_T$.
3. Obtain an candidate configuration $C_{c0} = C_c + S \cdot \boldsymbol{D}$.
4. Find a foldable configuration C_f by using C_{c0} as an initial state.
5. Determine whether C_f is valid:
    - if yes, $C_c = C_f, W = W + \Delta W$
    - if no, $W = W - \Delta W$
6. Determine whether reaching maximum iteration steps or C_c approaches C_T close enough.
    - if yes, terminate the procedure and and output the configuration
    - if no, repeat the process from Step 1

To find the folding configuration starting from the initial state C_{c0}, we need to check (2.7.1) to minimize the objective function

$$F(C) = |\prod_{j=1}^{n_c} M((i,j)) - I| \tag{24}$$

Self-intersection can be avoided by limiting all folding angles within $[-\pi, \pi]$. this constraints is checked in Step 5 to keep the configuration valid.

2.7.2 Fabric Folding through Spring-mass Model

The fabric surface is soft and deformable. Therefore we use origami folding as the pre-processing step for the canopy folding and apply the spring-mass model to the surface after the crease pattern has been created. Another difference between simple paper folding and the folding of a triangulated surface is that during the origami paper folding, each surface piece is planar, and thus can be represented by a small number of points. For a fabric surface in our computation, we need to transform all the mass points in the triangle mesh. Figure 15 shows the folding of surface with a triangle mesh.

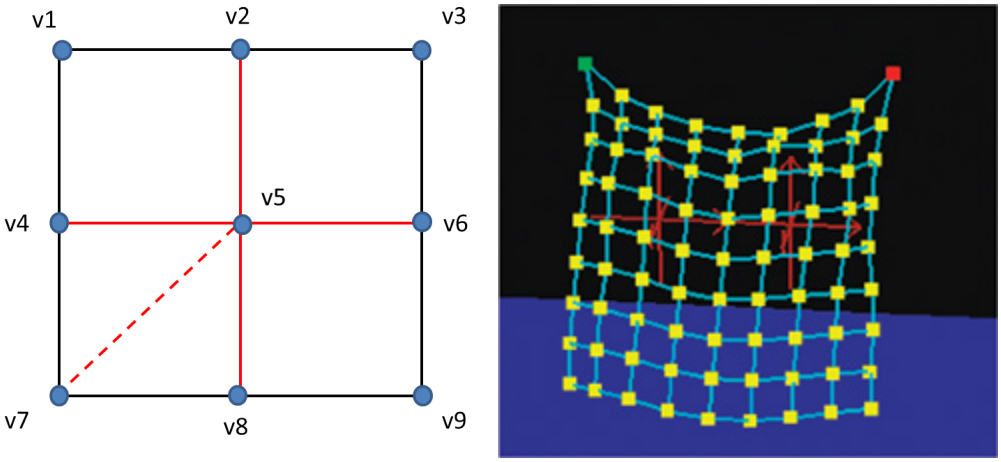


Figure 15: For origami folding, we need to work on only a few vertex points (left). The fabric surface is represeted by a triangle mesh, we need to transform all the vertices in the mesh (right).

A library is built to implement the folding algorithm. In the preprocessing stage, triangulation of the surface is carried out. We then specify which surface piece a mass point belongs to and determine its motion. The rigid origami algorithm is applied to fold the surface to the target shape. After this folding stage is completed, the surface will be sent a restoring precess and recover it's fabric property. This library can be used in the parachute simulation.

2.8 Parallelization

The *FronTier++* library offers functions for parallelized operations in initialization and front propagation. On this parallel platform, the computational domain is divided into partitions along each direction. A buffer interface is attached at the boundary of each subdomain. After propagation of the interface at each time step, the buffer surface is updated through the exchange of the buffer interface geometry and state data with the neighboring subdomains. For parachute simulation, we use PETSc to solve for the Navier-Stokes equation in both the advection and projection steps. The parallelization of ODE solver for the parachute canopy is more challenging and is parallelized on the GPU platform.

Global indexing is a new feature added to the *FronTier++* software library for fluid-structure interaction. Since the original *FronTier++* [11] deals with frequent surface mesh optimization and topological reconstruction, its parallelization is based on the floating point matching. The floating point matching is not completely reliable therefore more complicated algorithms were implemented as the reinforcement. For the fabric-like surface, especially when a spring-mass model is used, the inter-connectivity and proximity of the interface marker points are static. Therefore, global indexing is ideal for the parallel communication of the surface information and is now employed in the work. This new feature enables the

parallel communication of interface topology and geometry on a much more reliable and robust basis. It has greatly reduced the run-time interruption due to bugs and enhanced the efficiency in geometrical and topological matching at the buffer zones.

Graphics Processing Unit (GPU) computing [19] is to use the GPUs together with CPUs to accelerate a general-purpose scientific and engineering application. GPU computing can offer dramatically enhanced application performance by offloading computation-intensive portions of the programming code to the GPU units, while the remainder of the code still runs on the CPU. Joint CPU/GPU applications constitute a powerful combination because CPUs consist of a few cores optimized for serial processing, while GPUs consist of thousands of smaller, more efficient cores designed for massive parallel calculations.

Parachute type	CPU/GPU	Time(s)	Avg time per step(s)	Speedup
C9	CPU	2805.85	3.39	1.00
	GPU	131.90	0.16	21.2
G11	CPU	5101.47	5.41	1.00
	GPU	243.18	0.26	20.81
Intruder	CPU	1252.65	2.00	1.00
	GPU	69.67	0.11	18.18
T10	CPU	5540.02	5.99	1.00
	GPU	282.74	0.36	16.64
T11	CPU	6791.9	5.12	1.00
	GPU	352.07	0.29	17.66

Table 1: A comparison of computational time between different parachute type on CPU or GPU. The speed-up is calculated based on the computing time by CPU.

Our current strategy is that for a given parachute set, we collect the point data to a sever processor and then apply the GPU solver for each set. For multiple parachute sets, more server nodes are used. We carried out a series of tests of the GPU tests on the fabric simulator. The GPU code is implemented with CUDA library, which is a parallel computing platform created by NVIDIA. We compared the computational time of solving the spring model for different types of parachute with and without using the GPU device. The measurement of speed-up is shown in Table 1. The use of GPU device can achieve $16-21$ times speed-up for cloth simulation, thus can match up to 16 CPU processors (subdomains) for the fluid solver. Figure 16 is an example of parallel simulation of a three-G11 cargo parachute set in its inflation and descending process. In this simulation, total of 16 CPU cores are used and the spring-mass solver of the three canopy are computed by the GPU units.

2.9 Migration to Super Computer

The parallelization enables us to migrate the parachute code from workstations to cluster, with the goal that the code can eventually run efficiently on DOD's high performance com-

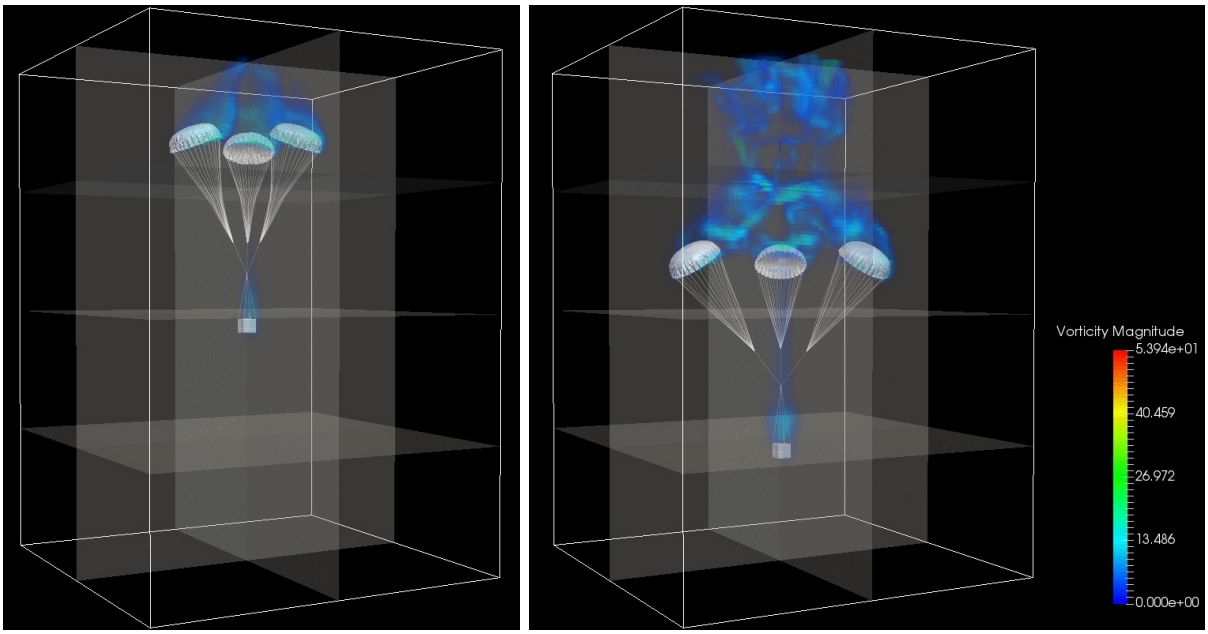


Figure 16: Parallel simulation of multi-parachute descending with a cubic cargo. The simulation is carried out on the parallel cluster with 16 subdomains. The neighboring subdomains are communicated via MPI while the spring model is solved using the GPU units. The numerical algorithm is capable of running on high performance parallel computer with large number of processors but requires further tuning on efficiency and load balance.

puters. Several experiments have been carried out to investigate the performance of our parallel computing platform: *FronTier++* , whose incompressible fluid solver relies on the parallel performance of external packages (such as PETSc and HYPRE). We carried out two sets of comparison tests on the strong and weak scalings of the PETSc package.

In the first test, a two dimensional Poisson equation was solved using the PETSc KSP solver with HYPRE as the preconditioner. To test the strong scaling, we fix the domain size to be 4096×4096 while continually doubling the number of processors from 1 to 1024. The scaling results are summarized in Table 2 and illustrated by left plot of Figure 17. The result showed that the super computer has a wider range of linear scaling than the Linux cluster and workstations. It also suggests that the speed-up becomes slower when the CPU cores are fully occupied. We interpret this as the result of the limited bandwidth between the core memory and the CPU, a conclusion agreed by the PETSc developers.

The second experiment is to test the weak scaling of *FronTier++* itself by solving the 2-D Riemann problem (no external package is used) through the MPI library. The base size for each processor is set to be 100×100, 200×200, 400×400 and 800×800, respectively. The efficiency is measured by T_1/T_N, where N is the number of processors. The results are displayed by the right plot of Figure 17 and Table 3. It showed that we can achieve the efficiency over 50% for up to 256 cores when the base size is smaller than 400×400 while the

efficiency decreases to 20% when the base size is 800×800. The scaling we have achieved is far from ideal, we are still in the process of tuning the program for better parallel efficiency.

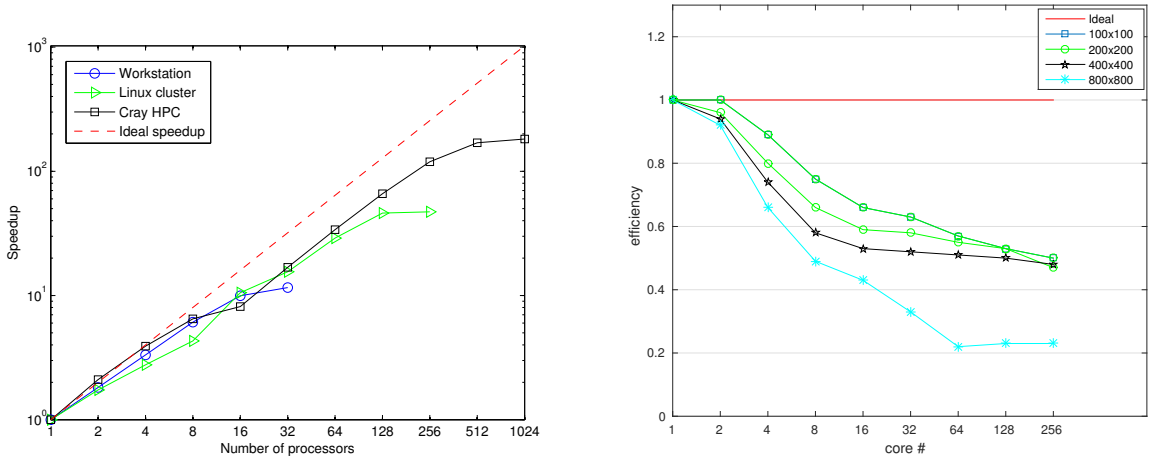


Figure 17: The left figure shows the speed-up of the linear solver for the 2D Poisson equation with mesh size (4096×4096) on different computer platforms: workstation, cluster and the DOD HPC. The right figure shows the weak scaling on the test of 2d Riemann problem. The efficiency is calculated with T_1/T_N, where N is the number of processors used. The results showed that 50% efficiency is achieved when mesh size per processor is smaller than 400×400.

Number of processors	1	2	4	8	16	32	64	128	256	512	1024
Time on workstation (s)	136.8	88.6	48.4	27.7	17.0	14.6	-	-	-	-	-
Time on linux cluster (s)	129.1	87.6	54.9	37.1	15.1	10.3	5.8	3.6	3.7	-	-
Time on HPC (s)	78.1	56.7	31.5	20.9	17.7	10.0	5.3	3.1	1.1	1.3	1.2

Table 2: Summary of the computational time in the strong scaling test on workstation, Linux cluster and CRAY HPC by solving the 2D Poisson equation.

3 Simulations on Different Scenarios

We present several numerical simulations to demonstrate the capability of the numerical model on various scenarios of the parachute system. These simulations include the coupled solution of the full parachute system to show some of the properties that may help in understanding and design of the parachute system, such as the porosity control, the wake effect of parachutist, and the risk of uneven and breaking suspension lines. We will also show the resolution of different types of collisions that can occur in the parachute simulation. It should

Size per proc	num of procs	1	2	4	8	16	32	64	128	256
100×100	time (s)	40	40	45	53	61	64	70	75	80
	efficiency	1.00	1.00	0.89	0.75	0.66	0.63	0.57	0.53	0.50
200×200	time (s)	85	89	106	129	145	149	154	159	181
	efficiency	1.00	0.96	0.80	0.66	0.59	0.58	0.55	0.53	0.47
400×400	time (s)	401	427	544	695	762	773	783	809	838
	efficiency	1.00	0.94	0.74	0.58	0.53	0.52	0.51	0.50	0.46
800×800	time (s)	2081	2263	3161	4244	4839	6334	9282	8865	9114
	efficiency	1.00	0.92	0.66	0.49	0.43	0.33	0.22	0.23	0.23

Table 3: Weak scaling test on 2D Riemann problem by continually doubling the mesh size per processor from 100×100 to 800×800 and doubling the number of cores.

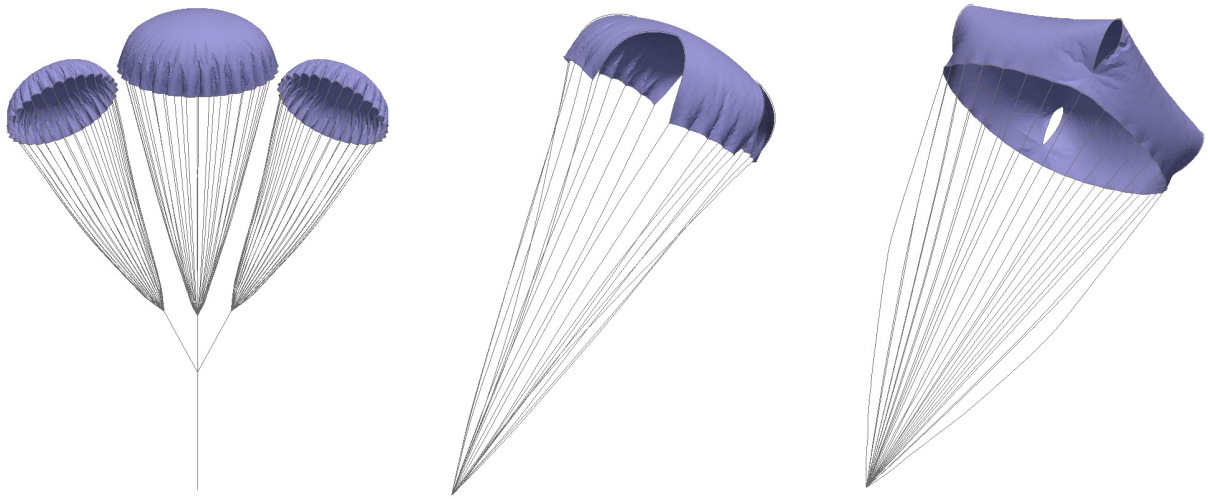
be mentioned that some of these simulation results are predictive and will need experimental validation.

3.1 Simulation of Different Types of Parachute

With our dual-stress spring-mass model and the numerical methods described in previous sections, we initialize different types of parachutes on the *FronTier++* platform. Figure **??** shows three types of fully inflated parachute canopies.

Our simulations shows the velocity pattern in the wake of parachutist and the canopy. Figure **??** shows the cross-sectional velocity magnitude at three different times. In this simulation, a C-9 personnel parachute with its nominal diameter $8.53m$ is placed in the computational domain of $16m \times 16m \times 36m$. The boundary conditions include a constant upward inflow velocity of $3m/s$, an upper outlet with constant pressure $p = 0$ and the periodic boundaries for the four horizontal sides of the domain. The mass of the payload, which is represented as a mass point here, is $100kg$.

Uneven or broken suspension lines are scenarios which can have adverse effect on the aerodynamic movement of the parachute. Here, we demonstrate two cases in the numerical wind tunnel test of the C-9 personnel parachute in a $12m \times 12m \times 24m$ computational domain. The boundary conditions are the same as above. In the left plot of Figure **??**, 10 of the suspension lines have the equilibrium length $0.3m$ shorter than the standard length of $7m$. In the right plot of Figure **??**, 10 selected suspension lines break at $t = 5.0s$. It is observed that in the first case, the parachute canopy surface becomes unstable, leading to an angled deployment. In the second case, the parachute canopy experiences a sudden and strong force from the side in which the suspension lines are broken and slide quickly to the direction opposite to the side of the broken lines.



(a) Three G-11 cargo parachute cluster

(b) Cross parachute

(c) T-11 personnel parachute

Figure 18: Three different types of parachute canopies simulated in *FronTier++* with our spring-mass model. Given parameters of a parachute, such as shape, size, number of chords, *FronTier++* has robust functions to generate them.

3.2 Effect of Porosity on Parachute Canopy

We present the example which shows the effect of porosity on the parachute canopy in a wind tunnel test. In this simulation, a G-11 cargo parachute with $10m$ nominal diameter is initialized and the load point is fixed. The computational domain is set to be $14m \times 14m \times 40m$ with constant velocity of $3m/s$ upward at the inlet, the outflow boundary at the top has constant pressure $p = 0Pa$, and the four horizontal sides are set for the periodic boundary conditions. Two cases, one with zero porosity and the other one with porosity coefficients $\alpha = 6.7kg \cdot m^{-1} \cdot s^{-1}$ and $\beta = 3.1kg \cdot m^{-2}$ are simulated.

As shown in Figure **??**, the parachute canopy in both cases fully opens. However, when porosity is added to the canopy, the vorticity field is more symmetric around the parachute canopy and the wake flow is less turbulent. Therefore porosity contributes to the stability of the parachute system. A fine tuned material porosity tends to modulate balance the drag force and adds to the stability of the canopy motion, thus optimizes the design of the parachute system.

3.3 Wake Effect of the Parachutist

By adding the rigid body component to the spring-mass model, we take into account the turbulent effects caused by the wake of the parachutist. In this simulation, we compute the flow around a C-9 personnel parachute (without vent) with its nominal diameter of $8.53m$. The computational domain is $16m \times 16m \times 36m$ with constant upward inflow velocity of

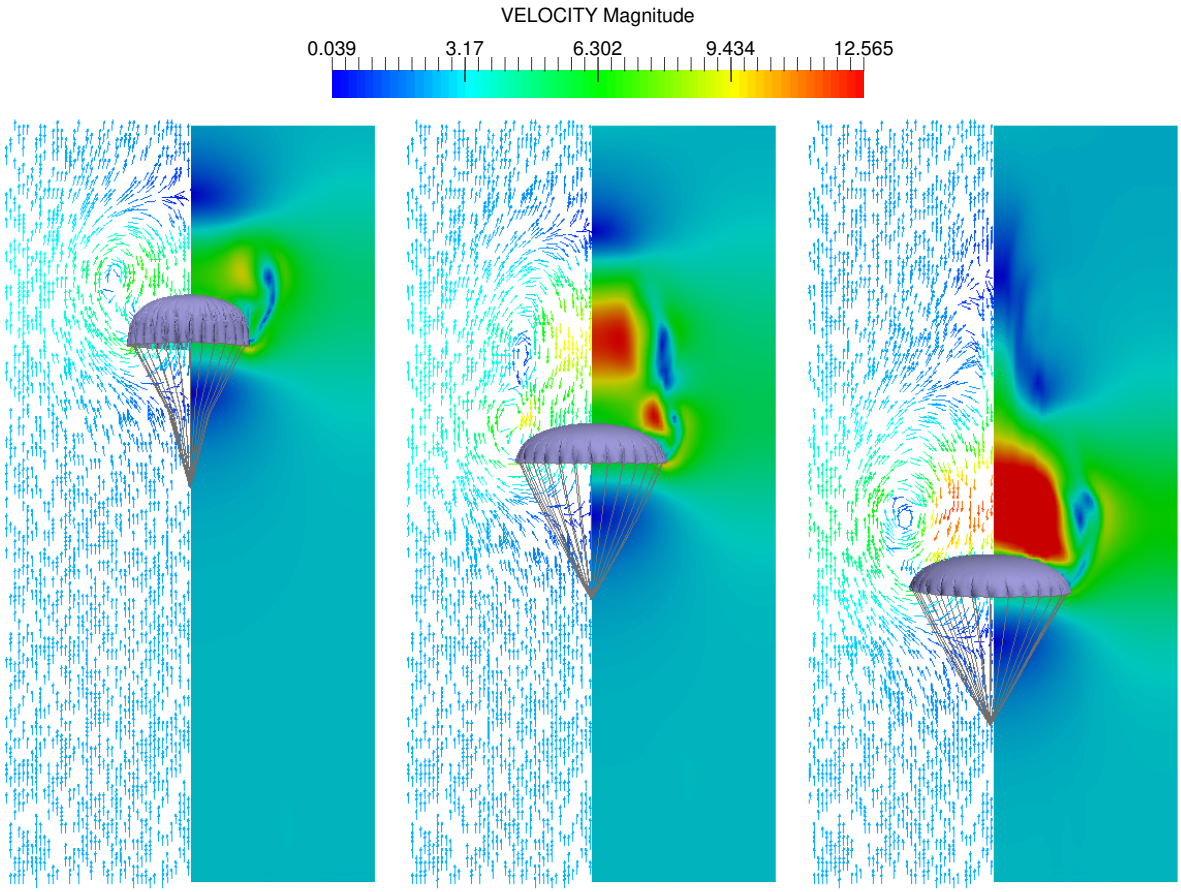
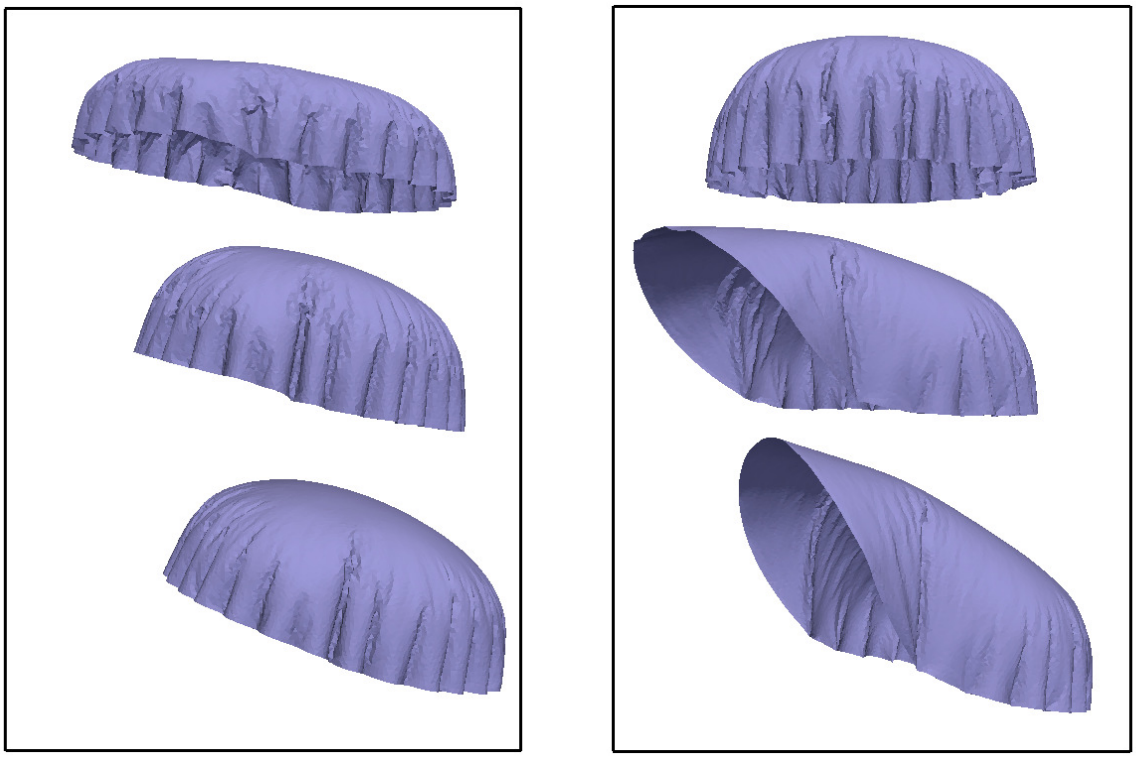


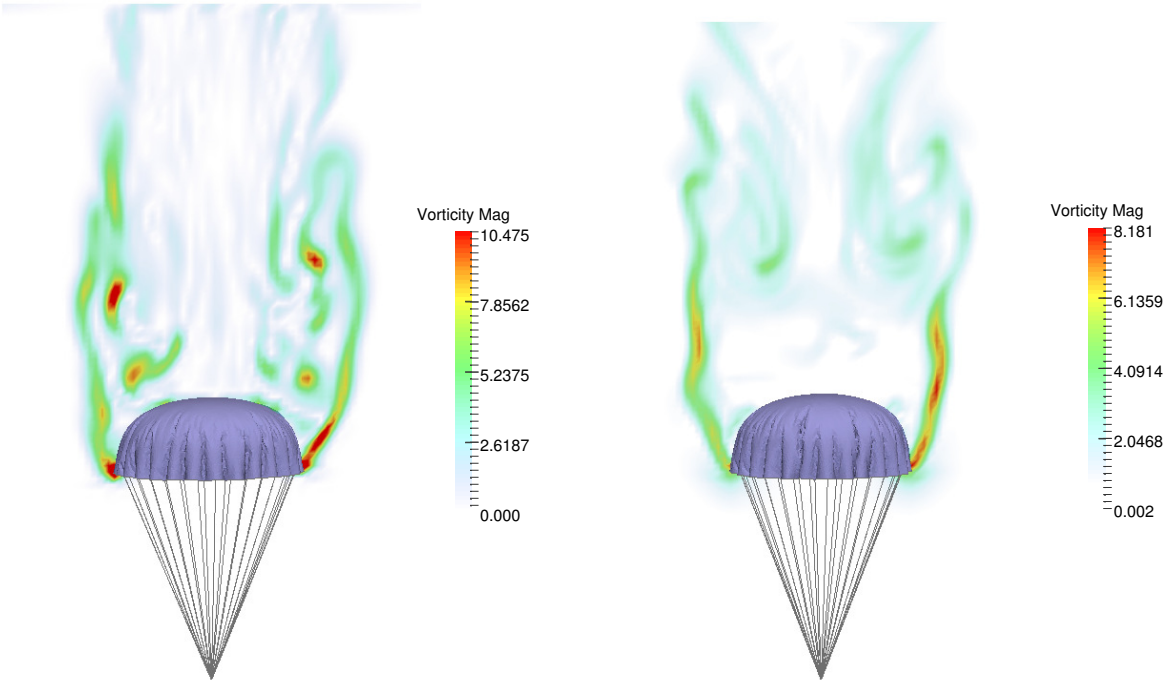
Figure 19: The cross-sectional velocity profiles around C-9 personnel parachute during the descent process, at time $t = 1.2s$, $t = 2.0s$ and $t = 3.2s$, respectively. In each plot, the left half shows the velocity direction by arrow and the right half shows the velocity magnitude by color, where red is large and blue is small.



(a) Uneven suspension lines

(b) Breaking suspension lines

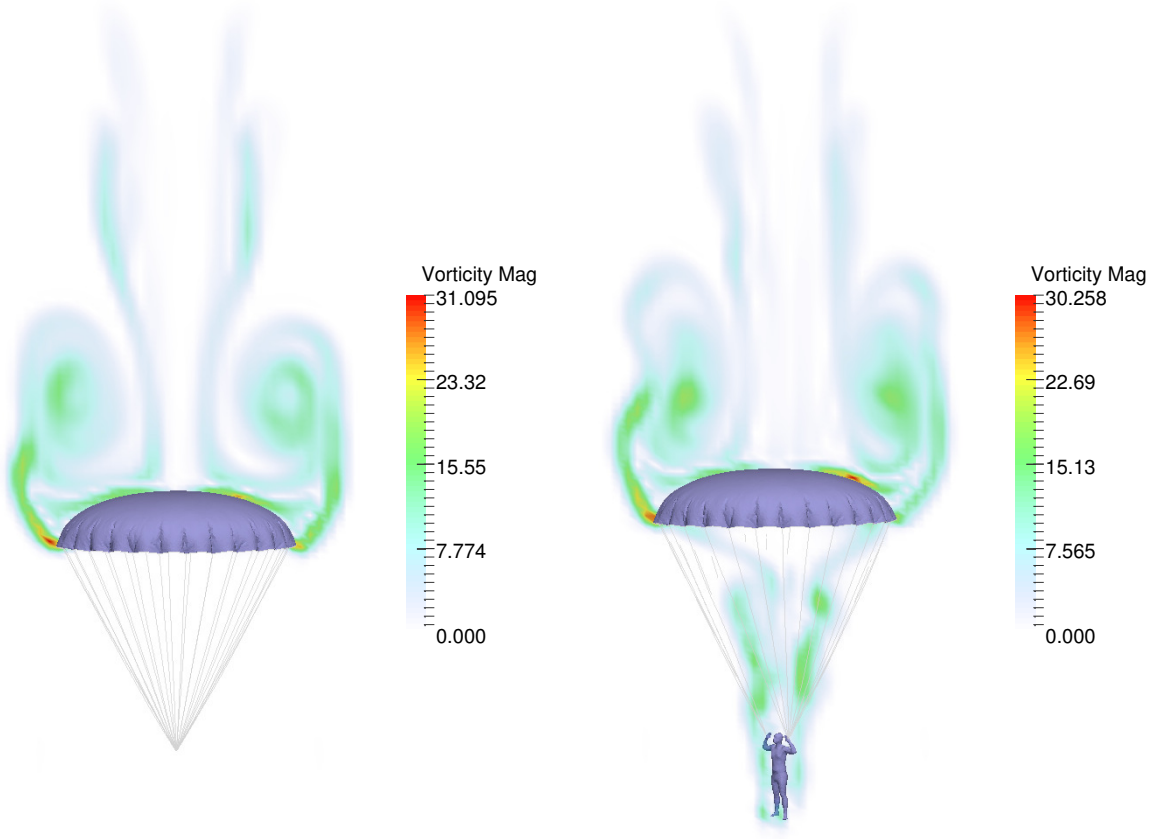
Figure 20: The canopy surface of the C-9 personnel parachute at different time. Left: 10 suspension lines are selected to have an equilibrium length shorter than others. Right: 10 suspension lines are selected to break at time $t = 5.0s$.



(a) Vorticity field without porosity

(b) Vorticity field with porosity

Figure 21: Comparison of the cross-sectional vorticity magnitude field at $t = 10.0s$. In the left figure, the canopy surface is non-porous and no flow goes through it. The right plot is the case with porosity coefficients $\alpha = 6.7kg \cdot m^{-1} \cdot s^{-1}$ and $\beta = 3.1kg \cdot m^{-2}$.



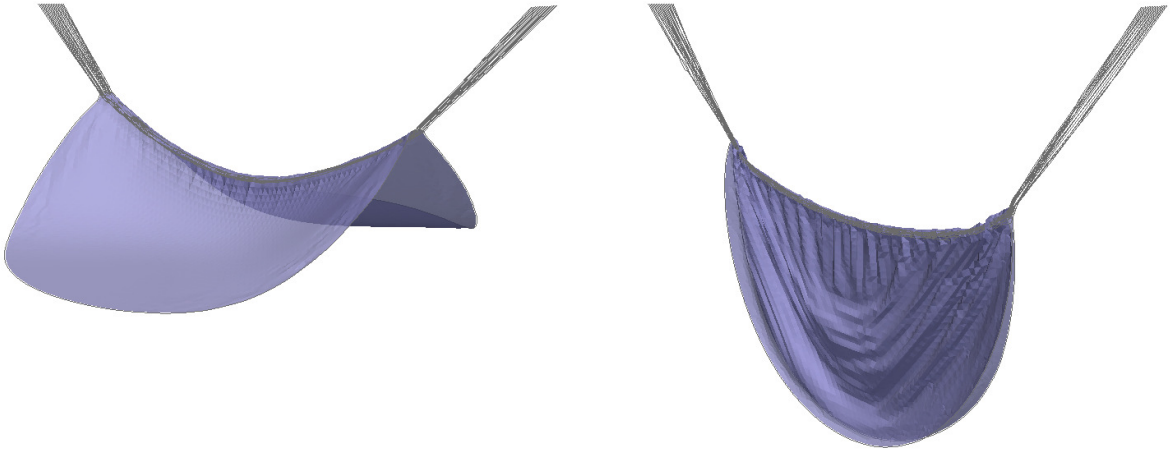
(a) Vorticity field without parachutist

(b) Vorticity field with parachutist

Figure 22: Comparison of the cross-sectional vorticity magnitude field. In the left figure, the parachutist is a mass point, while in the right figure, it is a rigid body with finite volume.

$3m/s$ at the lower side, outflow pressure $p = 0$ at the upper side, and periodic boundaries for the four horizontal sides. The weight of the parachutist is set to be $100kg$.

Figure **??** shows that a turbulent wake appears as the fluid flow passes the parachutist. The descent speed of the parachute system is slightly smaller in comparison with the simulation in which the payload is a point weight. We believe that this is due to the interaction between the parachutist and the fluid flow, which generates vorticity and increases the upward impulse on the parachute canopy. Conceptually, the form drag, caused by the pressure mismatch acting on the frontal area of the parachute presented to the flow, is greater in the laminar case than in the turbulent case. Although the frictional drag of the turbulent case is higher, the form drag is the dominant term in both cases. Therefore we believe the results are reasonable.



(a) Early stage: no string clustering (b) Later stage: strings are clustered

Figure 23: Two stages in the process of a circular cloth falling onto a collection of elastic strings. In the left figure, there are collisions between cloth and strings, while in the right figure, the are collisions between the strings and self-collision of cloth.

3.4 Simulations of Fabric Surface and String Chord

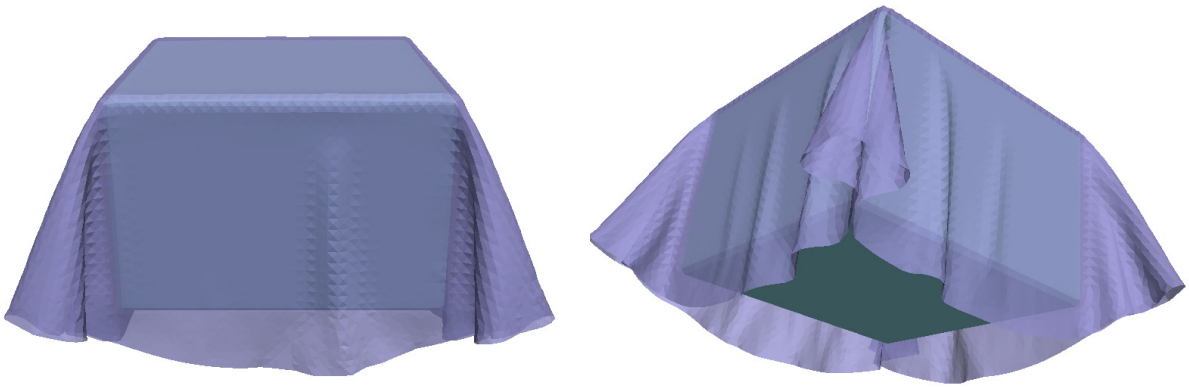
Coupling with the rigid body dynamics and the dual-stress soring-mass model, we study the performance of the collision algorithm under different circumstances.

3.4.1 Case 1: Cloth and Strings

The first example involves the collision between a fabric surface and elastic strings. Initially, a flat circular fabric surface with $0.3m$ radius is placed right above a set of straight and parallel strings with their ends fixed. All the strings have the same equilibrium length of $0.8m$. Due to the gravity, the fabric surface falls on the strings and collisions occur. The fabric surface then begins to fold and the strings are clustered. After that the strings will also collide with each other. Finally, two folded parts of the fabric surface meet at the bottom and collide with each other. Figure **??** shows how the algorithm resoles three different types of collisions.

3.4.2 Case 2: Collision Between Rigid Body and Fabric Surface

We present two examples to show the collision between rigid body and fabric surface. In the case shown by Figure **??**, a circular fabric surface with $0.4m$ radius falls to a static cuboid object, whose side length is $0.3m$. Initially, the fabric surface is $0.05m$ above the object. The bending force is added as described in Sec. **??**. It highly reduces the number of collision pairs and forms the billows at four corners.



(a) Front view of the cuboid (b) Side view of the cuboid

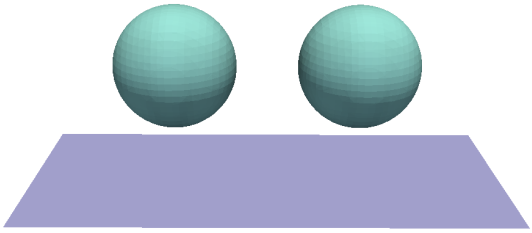
Figure 24: The front view and the side view of the cloth falling onto static objects. Benefited from the bending effect mentioned at the end of the spring-mass model, billows appear at the corners.

Figure **??** is the case of two balls falling to a fabric surface with a length of $0.8m$ and a width of $0.33m$. Two balls both have a radius of $0.1m$ and are initially $0.1m$ above the fabric surface. The left and the right sides of the fabric surface are fixed. At first, the two balls collides with the fabric surface, and only the elastic rigid body and fabric surface collision occurs. The two balls are then pushed towards each other by the elastic force from the fabric surface and collide with each other. The collision between the two balls is set to be fully inelastic, which means the restitution coefficient is $cr = 1.0$, therefore they keep in contact with each other and settle at the center of the fabric surface.

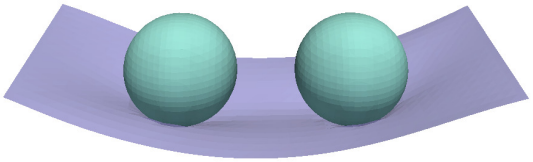
4 Other Research Development

We continued our development of the *FronTier++* library as an open source software for the computational science community. This includes the enhancement of the topological merge and bifurcation, its reliability and efficiency, and the implementation of new algorithms such as searching and distance calculation.

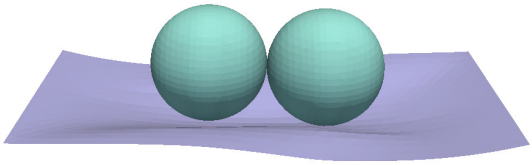
Among the new applications we have developed are the airbag simulation module, the compressible fluid solver application for small arms simulation and supersonic parachute platform. These applications can be used as validation tools for the *FronTier++* library. They are also potential application tools for the Army and other federal research programs such as those of NASA.



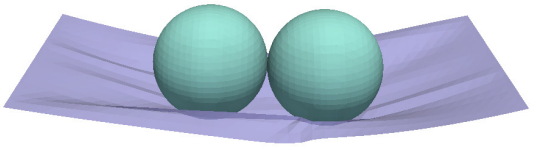
(a) Initial stage, two balls are at some height.



(b) First stage: two balls collides with cloth.



(c) Second stage: two balls meet in the middle.



(d) Third stage: all structures collide together.

Figure 25: Two balls falling on to cloth whose two sides are fixed.

4.1 High Fidelity Compressible Fluid Solver

We have implemented and tested two high fidelity compressible fluid solvers, the weighted essentially non-oscillatory (WENO) scheme [16, 25] and the finite element discontinuous Galerkin (DG) scheme [7]. We have computed various benchmark test problems including one and two dimensional Riemann problems, Rayleigh-Taylor and Richtmyer-Meshkov instability problems, and the fluid-structure interaction problems. Figure 26 shows the simulation of a supersonic gun-fired bullet traveling in the air using the fifth order WENO scheme. These high fidelity solvers enable us to extend the research into more interesting problems. For example, we have written a proposal to NASA for the study supersonic parachute deceleration system used by the re-entry vehicles.

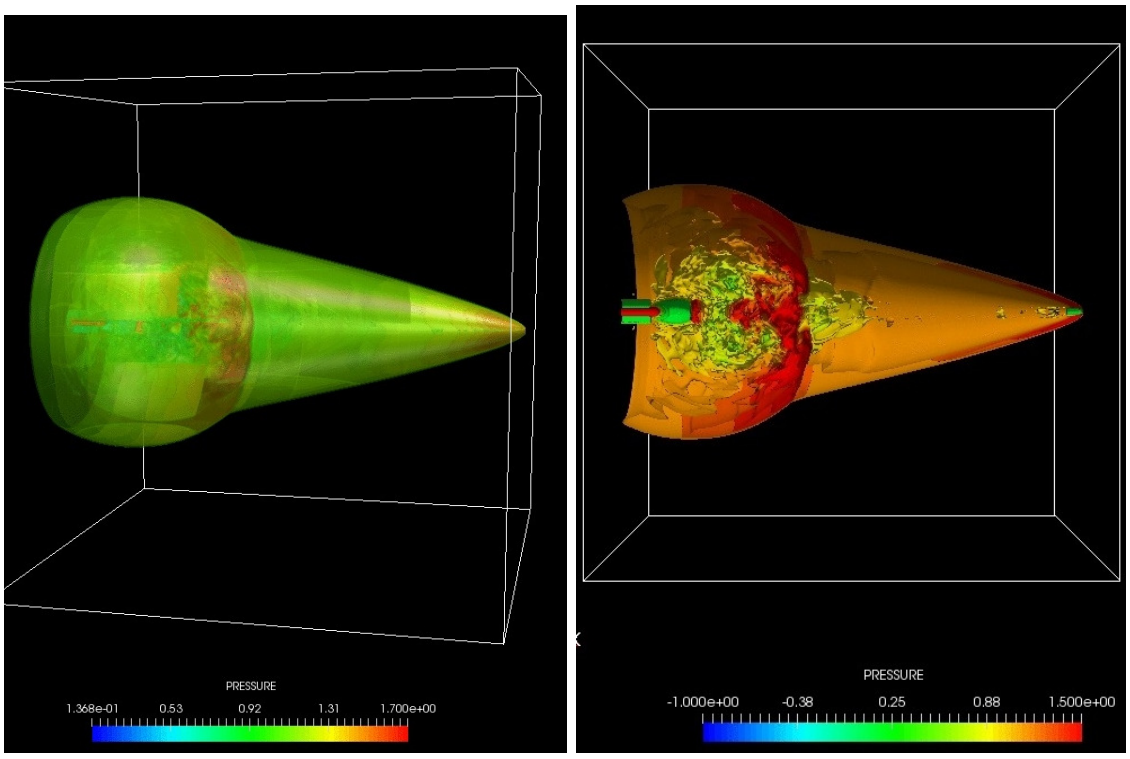


Figure 26: Simulation of a gun-fired supersonic bullet using the fifth order WENO scheme with high resolution on shock front (pressure and cross-sectional pressure).

4.2 Airbag Modeling and Simulation

Numerical simulation is an important method to do in-depth study on the airbag performance on its interaction with the occupants in a collision. The results may provide references for airbag design and deployment. Our spring-mass fabric model and the impulse method can be applied to the airbag simulation if it is coupled with the CAD software for vehicles. Figure 27

shows the simulation of airbag inflation using our dual-stiffness spring-mass model coupled with the impulse method.

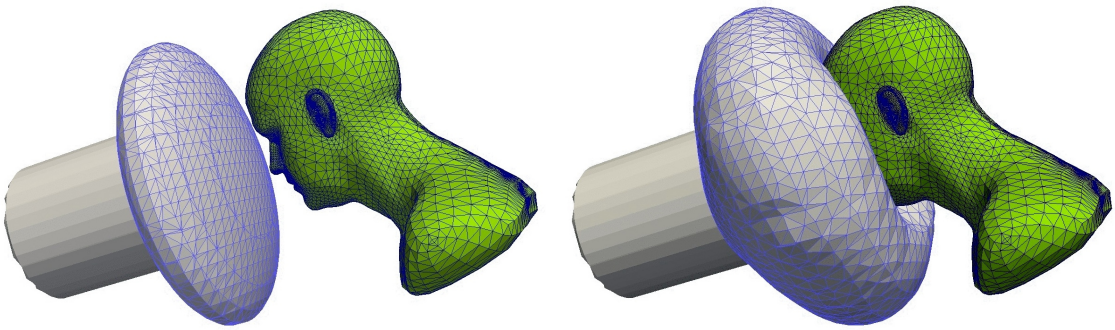


Figure 27: Simulation of airbag inflation using the dual stiffness spring-mass model and the *FronTier++* library.

4.3 Conservative Phase Transition

Deposition (cristalization) and dissolution (leaching) have many industrially important applications in chemical and nuclear engineering. We have continued to develop a fully conservative tracking algorithm for such problem. Figure 28 shows the simulation of dissolution of a chemical sample in the convecting fluid flow. This work is in collaboration with Dr. Valmor de Almeida at the Oak Ridge National Laboratory. Recently we have extended the conservative algorithm into three dimensions.

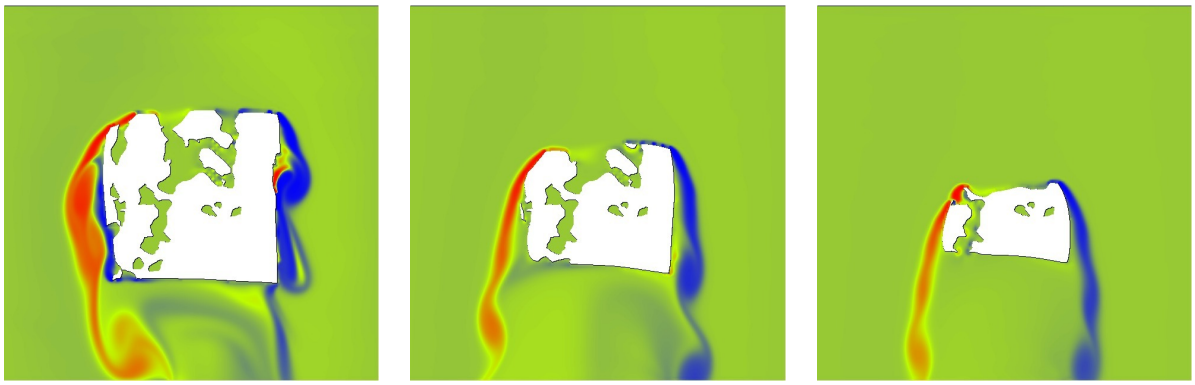


Figure 28: Application of the fully conservative front tracking method on the dissolution simulation. The color shows the vorticity of the fluid flow carrying away the solute dissolved from the solid chemical sample.

4.4 Cloud Entrainment for Climate Modeling

We have developed a DNS simulation module on the *FronTier++* platform for the study of cloud entrainment in climate modeling. This project is in collaboration with Drs. Yangang Liu and Robert McGraw at the Brookhaven National Laboratory. This work will be presented at the DOE ASCR-BER workshop on supercomputing for climate science in September. Figure 29 shows one case of the simulation in which cloud particles are clustered at the edge of vortices.

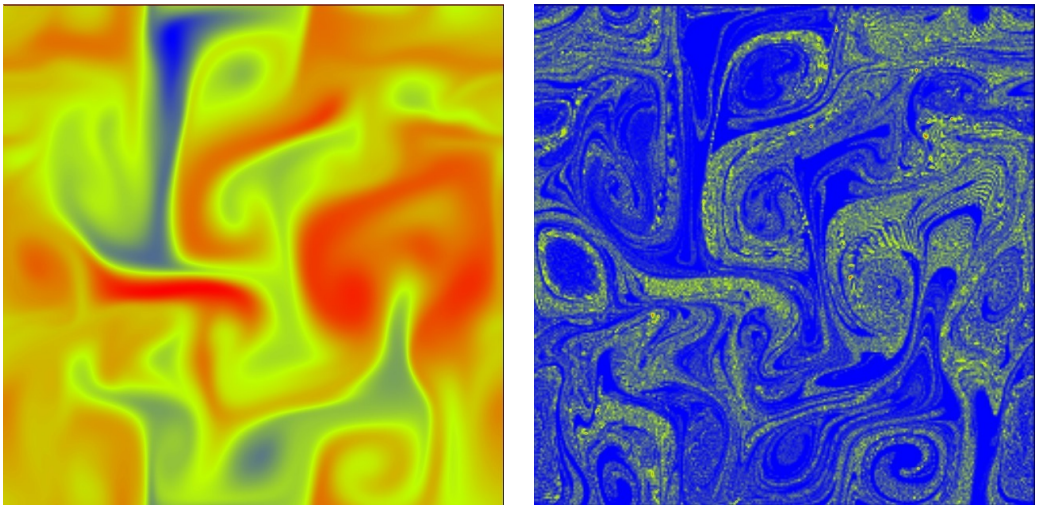


Figure 29: Simulation of cloud particle entrainment. The left plot is a snapshot of air vorticity and the right plot is the particle concentration. The simulation shows that the cloud particles are clustered at the edge of vortices.

5 Outreach of the Program

5.1 Collaboration with Army Scientist

Our project has been constantly advised by Army scientist Dr. Richard Charles of the Airdrop Technology Team at the US Army Natick Research Development and Engineering Center (NSRDEC) with his expertise in modelling, experiment and field experience on the parachute delivery system. Our first joint paper [23] was published by Journal of Fluid and Structure in 2015. Our second joined paper with Dr. Charles [34] has also been accepted for publication by the AIAA Journal.

5.2 Software Distribution

Our *FronTier++* code has been requested and distributed to the following DOE laboratories and academic institutions:

(1). Brookhaven National Laboratory, Department of Environmental and Climate Science.

(2). Oak Ridge National Laboratory, Department of Material Science.

(3). University of Connecticut Medical Center.

(4). University of Waterloo, Canada.

5.3 Impact on Education and Training

Supported in part by the ARO grant and other resources, our program has produced five graduates with Ph. D. degrees and one with master's degree. Among them, two are American students while others are international students later employed by American universities and high-tech companies including Texas A&M University, Google, Bloomberg, and Knight Capital Group Inc. One former international student, Eric Rathnayke, joined the US Army on the MAVNI (Military Accessions Vital to the National Interest) program.

Six high school students have been mentored by our research group under the ARO HSAP program. Four of them later were admitted by universities and continued in STEM field.

Our computer equipments, including workstations and the parallel cluster, have also been shared with two junior faculty members outside our research group. This includes Dr. Pei Fen Kuan, an assistant professor of the Stony Brook University working on the statistical and computational methodologies to facilitate the analysis, integration and interpretation of high-throughput epigenomics data arising from micro-arrays and next generation sequencing platforms. Another faculty member is Dr. Jennie D'Ambroise, an assistant professor of SUNY South Hampton whose research involves the study of numerical, analytic, and semi-analytic solutions of the Schrödinger equations with various types of nonlinearities and potential functions.

References

[1] T10 folding technical manual unit and direct support (ds) maintenance manual. 1988.

[2] B. S. Baldwin and H. Lomax. Thin layer approximation and algebraic model for separated turbulent flow. *AIAA 16th Aerospace Sciences Meeting*, January 1978.

[3] Sarah-Marie Belcastro and Thomas C Hull. A mathematical model for non-flat origami. In *Origami3: Proc. the 3rd international meeting of origami mathematics, science, and education*, pages 39–51, 2002.

[4] D. Brown, R. Cortez, and M. Minion. Accurate projection method for the incompressible Navier Stokes equations. *J. Comp. Phys.*, 168:464–499, 2001.

[5] Han Cheng, Li Yu, and ZW Yin. A new method of complicated folded fabric modeling. *Journal of Harbin Institute of Technology*, 19(2):43–46, 2012.

[6] A. J. Chorin. Numerical solution of the Navier Stokes equations. *Math. Comp*, 22:745–762, 1968.

[7] C.-S Chou, C.-W Shu, and Y. Xing. Optimal energy conserving local discontinuous galerkin methods for second-order wave equation in heterogeneous media. *Journal of Computational Physics*, 272:88–107, 2014.

[8] Herve Delingette. Triangular springs for modeling nonlinear membranes. *IEEE Transactions on Visualization and Computer Graphics Volume 14 Issue 2*, pages 723–731, March 2008.

[9] Erik D Demaine and Martin L Demaine. Recent results in computational origami. In *Origami3: Third International Meeting of Origami Science, Mathematics and Education*, pages 3–16, 2002.

[10] Erik D Demaine, Martin L Demaine, and Joseph SB Mitchell. Folding flat silhouettes and wrapping polyhedral packages: New results in computational origami. In *Proceedings of the fifteenth annual symposium on Computational geometry*, pages 105–114. ACM, 1999.

[11] Jian Du, Brian Fix, James Glimm, Xicheng Jia, Xiaolin Li, Yunhua Li, and Lingling Wu. A simple package for front tracking. *Journal of Computational Physics*, 213(2):613–628, 2006.

[12] James M. Gere. *Mechanics of materials, sixth edition*. Bill Stenquist, 2004.

[13] Shinya Hayashi. Jfold-introducing a new simulation-based airbag folding system for ls-dyna. 2013.

[14] Shinya Hayashi and Richard Taylor. Simulation-based airbag folding system jfold version 2: New capabilities and folding examples. 2014.

[15] Thomas Hull. On the mathematics of flat origamis. *Congressus numerantium*, pages 215–224, 1994.

[16] G. Jiang and C.-W. Shu. Efficient implementation of weighted ENO schemes. *J. Comput. Phys.*, 126:202–228, 1996.

[17] Hamid Johari and Kenneth J Desabrais. Vortex shedding in the near wake of a parachute canopy. *Journal of Fluid Mechanics*, 536:185–207, 2005.

[18] Joung-Dong Kim, Yan Li, and Xiaolin Li. Simulation of parachute fsi using the front tracking method. *Journal of Fluids and Structures*, 37:100–119, 2013.

[19] David B Kirk and W Hwu Wen-mei. *Programming massively parallel processors: a hands-on approach*. Morgan Kaufmann, 2010.

[20] Dmitri Kuzmin, Otto Mierka, and Stefan Turek. On the implementation of the k-epsilon turbulence model in incompressible flow solvers based on a finite element discretisation. *International Journal of Computing Science and Mathematics*, 1(2):193–206, 2007.

[21] ADRIAN J LEW, Gustavo C Buscaglia, and Pablo M Carrica. A note on the numerical treatment of the k-epsilon turbulence model. *International Journal of Computational Fluid Dynamics*, 14(3):201–209, 2001.

[22] Xu-Dong Liu, Ronald P Fedkiw, and Myungjoo Kang. A boundary condition capturing method for Poisson's equation on irregular domains. *Journal of computational Physics*, 160(1):151–178, 2000.

[23] Q. Shi, D. Reasor, Z. Gao, X.-L. Li, and R. D. Charles. On the verification and validation of a spring fabric for modeling parachute inflation. *Journal of Fluids and Structures*, 58:20–39, 2015.

[24] Qiangqiang Shi, Daniel Reasor, Zheng Gao, Xiaolin Li, and Richard D Charles. On the verification and validation of a spring fabric for modeling parachute inflation. *Journal of Fluids and Structures*, 58:20–39, 2015.

[25] C.-W Shu. High order weno and dg methods for time-dependent convection-dominated pdes: a brief survey of several recent developments. *Journal of Computational Physics*, 316:598–613, 2016.

[26] Tomohiro Tachi. Simulation of rigid origami. *Origami*, 4:175–187, 2009.

[27] Anand S Tanavde, Himanshu Khandelwal, David Lasry, Xiomin Ni, Eberhard Haug, Jutta Schlosser, and Pradeep Balakrishnan. Airbag modeling using initial metric methodology. 1995.

[28] Richard Taylor and Shinya Hayashi. Using jfold & ls-dyna to study the effects of folding on airbag deployment. 2015.

[29] David C Wilcox et al. *Turbulence modeling for CFD*, volume 2. DCW industries La Canada, CA, 1998.

[30] Zhonghua Xi and Jyh-Ming Lien. Folding rigid origami with closure constraints. In *ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages V05BT08A052–V05BT08A052. American Society of Mechanical Engineers, 2014.

[31] Victor Yakhot and Leslie M Smith. The renormalization group, the epsilon-expansion and derivation of turbulence models. *Journal of Scientific Computing*, 7(1):35–61, 1992.

[32] Yanan Zhan, Li Yu, Xue Yang, and Han Cheng. Initial stress correction method for the modeling of folded space inflatable structures. *Aviation*, 18(4):166–173, 2014.

[33] Jinhuan Zhang, Chunsheng Ma, Yuanli Bai, and Shilin Huang. Airbag mapped mesh auto-flattening method. *Tsinghua Science & Technology*, 10(3):387–390, 2005.

[34] Gao Zheng, Charles Richard D, and Li Xiaolin. Numerical modeling and simulation of flow through porous fabric surface. *AIAA Journal*, 2016. accepted for publication.